HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

MASTERARBEIT

Herr
Christoph Leberecht

Basic discrete concepts for simulating natural systems

An approach to describe diffusion and chemical kinetics using a combination of graphs and cellular automata

2015

Fakultät **Mathematik/Naturwissenschaften/Informatik**

MASTERARBEIT

Basic discrete concepts for simulating natural systems

An approach to describe diffusion and chemical kinetics using a combination of graphs and cellular automata

Autor:
Christoph Leberecht

Studiengang:
Molekularbiologie/Bioinformatik

Seminargruppe:
MO13-w1b

Erstprüfer:
Prof. Dr. Dirk Labudde

Zweitprüfer:
Florian Heinke, M.Sc.

Mittweida, August 2015

Bibliografische Angaben

Leberecht, Christoph: Basic discrete concepts for simulating natural systems, An approach to describe diffusion and chemical kinetics using a combination of graphs and cellular automata, 73 Seiten, 25 Abbildungen, 5 Tabellen, Hochschule Mittweida, University of Applied Sciences, Fakultät Mathematik/Naturwissenschaften/Informatik

Masterarbeit, 2015

Referat

Verschiedenste Methoden wurden bereits genutzt, um natürliche Phänomene und zelluläre Funktionen zu beschreiben. Ein Großteil dieser Methoden nutzt kontinuierliche Systeme in Verbindung mit Differenzialgleichungen. Basierend auf den Nachbarschaftsbeziehungen in Graphen und den Interaktionen in Zellulären Automaten, wurde ein mathematisches Modell konstruiert und als Programmierschnittstelle zur Verfügung gestellt. Dieser diskrete Ansatz, der sogenannte Graphen Automat, wurde genutzt, um Diffusion und die Kinetik chemischer Prozesse zu simulieren. Der Verlauf der Diffusion in zellulären Umgebungen konnte mit einer 20 prozentigen Abweichung, im Vergleich zu experimentellen Ergebnissen beschrieben werden. Die Abläufe verschiedener Reaktionen wurden simuliert und stellten sich als ebenso realistisch heraus, wie ihre kontinuierlichen Gegenstücke. Das vorgestellte Modell präsentiert sich als hoch skalierbarer und modularer Ansatz für die Simulation natürlicher Systeme.

A variety of methods have been used to describe natural systems and cellular functions. Most use continuous systems with differential equations. Based upon the neighbourhood relations in graphs and the complex interactions in cellular automata a mathematical model was designed and implemented as an application user interface. This discrete approach called graph automata was utilised to simulate diffusion processes and chemical kinetics. The progression of diffusion in cellular environments was described and resulted in a discrepancy of 20% in comparison to experimental results. Different chemical kinetics were simulated and found to be as accurate as their continuous counterparts. The proposed model appears to be a highly scalable and modular approach to simulate natural systems.

I. Contents

II. List of Figures

III. List of Tables

IV. Nomenclature

API	application programming interface
BCECF	2',7'-Bis-(2-Carboxyethyl)-5-(and-6)-Carboxyfluorescein
CCF	concentration collection function
CS	concentration set
DDS	discrete dynamic system
DE	differential equations
DFA	deterministic finite automaton
EGAC	extended graph automaton cell
FBA	fructose bisphosphate aldolase
FRAP	fluorescence recovery after photobleaching
GA	graph automaton
GAC	graph automaton cell
GUI	graphical user interface
ODE	ordinary differential equations
RI	reference implementation
RR	recurrence relations
SPE	systems of partially differential equations
STDF	solute transitional diffusion coefficient

V. Acknowledgement

For Marlene. For my family.

Firstly, I would like to express my sincere gratitude to my advisor Prof. Dr. rer. nat. Dirk Labudde for the continuous support throughout my master's study, for his patience, motivation, and immense knowledge.

I would also like to thank Florian Heinke, M.Sc. for his insightful comments and encouragement, but also for challenging my approaches, which broadened my view for other possibilities.

Additionally I especially like to thank Florian Kaiser, who supported me with his never-ending energy and assurance.

The guidance of my colleagues and foremost friends Sven Becker, Tommy Bergman, Sebastian Bittrich and Alexander Eisold helped me in all the time of research and writing of this thesis.

My sincere thanks also goes to Dr. rer. nat. Thorsten Liebers, for his critique, untiring support and patience in editing the manuscript.

Last but not the least, I would like to thank my family for supporting me spiritually and financially throughout writing this thesis and my life in general.

1 Introduction

> It may happen that small differences
> in the initial conditions produce very
> great ones in the final phenomena.
>
> *Henri Poincaré*
> *Science and Method 1903*

1.1 Motivation

It has often been stated that cellular automata are a way to simulate complex phenomena with a simple set of assumptions [Adami, 1998]. In many areas of science cellular automata are used to simulate complex and dynamic systems [Wolfram, 2002]. For example REINHARD KOENIG uses cellular automata to simulate the development of cities on a large scale [Koenig, 2012]. But small scale applications are equally as feasible. STANISLAW ULAM postulated the abstract idea of cellular automata while studying the growth of crystals. The scale of the atomic units in the automata should be chosen carefully, but can be chosen freely. In the field of cellular biology it has been suggested, that cellular automata (dynamic cellular automata) can be used to simulate chemical interactions and the progression of biochemical pathways [Wishart et al., 2005]. But until now, systems using cellular automata have been treated shabbily in the different fields of biology.

Differential equations (DE) seem to be the holy grail when trying to solve any problem that originates from a cellular environment. For instance ordinary differential equations (ODE) are used routinely to describe reactions, diffusion, or other processes that are in some kind dependent on a rate of change. Furthermore, systems of partially differential equations (SPE) can be used to model more complex and interdependent systems such as metabolism and cell signalling [Endy and Brent, 2001]. Nevertheless, not all SPE are solvable and not all processes can be described by DE. Sophisticated SPE can be be solved by powerful computational solvers, that are not always easy to adjust. DE in general lack the ability to include unexpected changes, once modelled, the behaviour of the curve is established and unable to respond to disturbances [Wishart et al., 2005]. The more aspects have to be included into the system, the more mathematical modelling has to be done, in order to capture all aspects that influence its behaviour. This provides a challenge even for scientists with a strong background in mathematics. A system that is based on DE can not easily be modified, because all equations in SPE have to be

coordinated and synchronized. It is out of question that such a well tuned system is reliable and can be solved at any point to gain information about the system at a critical time. Yet any modification to the system results in modifications to all equations [Brin and Stuck, 2002].

A modular system that allows to easily add and remove certain aspects of a model is an elegant way to analyse biological systems. A "prototype" of a model can be used as a foundation to experiment with different pathways, reaction types, and parameter sets, without the need to modify everything due to the recently added component.

Current methods to simulate cellular processes, including COPASI [Hoops et al., 2006], VCell [Slepchenko et al., 2003], and E-Cell [Ishii et al., 2004], rely on differential equations. VCell is probably the most used and refined simulation software for cellular simulation. The user presents the system different DE that are than combined to SPE and solved numerically [Slepchenko et al., 2003]. But the bypass is always taken via DE. Although DE are created to solve a system at any point in time, it is than solved at discrete time steps numerically. This procedure could be simplified with recurrence relations (RR), where the next state of a system is strictly defined in relation to the previous time step. If the time step was small enough in relation to the speed of the processes that are occurring, DE could be replaced with RR [Hütt, 2001].

In this work cellular automata have been chosen to build a concept that encapsulates an easily modifiable and scalable model to simulate natural systems. The cells of the automata and their neighbourhood relations between each other represent the spatial extend of the environment. The time component is represented by discrete time steps, where the next state of the system is calculated depending on the current time step. The goal is to make the simulation of a cellular system more vivid. Every components should be easily modifiable, even while a system is running, and the set up of the system should be straightforward. Many tools for data integration are helpful in this process, with the result that the user does not have to set up every parameter that is coupled to different chemicals or natural constants. Another important component is to go a step back from DE to RR. Many users may be hindered by seemingly difficult equations that are not convertible to DE but can be stated as RR.

1.2 Background

1.2.1 Systems biology

"Ask five different astrophysicists to define a black hole and you will get five different answers. But ask five different systems biologists and you will get ten answers" [Wanjek, 2011]. In general, it could be stated that systems biology is concerned with the mathematical modelling of complex biological systems. Because "biological systems" is a very broad term and there is a lot of interpretations. But to start with, it is necessary to provide the large amount of information about the chosen system in order to derive regularities. Therefore it is possible to attribute the emergence of systems biology to bioinformatics. With technologies like Next Generation Sequencing, yeast two-hybrid screening, and X-ray crystallography a lot of data is collected, and subsequent analyses are demanded. Primarily tools for statistical analyses, machine learning, clustering, and pattern recognition were created, to give the huge amounts of data structure. The first step was classification. How can nucleic acid sequences can be compared? If they were similar, could they code for similar protein structures? What can we learn from similar structures about the function and evolution of a protein? Those, and many more important questions drive bioinformatics towards an understanding of function of proteins and evolution of life. Systems biology now asks: How do those and other parts of biological systems interact and behave over time? The mentioned parts can be reachable on many different levels: From the behaviour of molecules in biochemical pathways and their transport, over the organisation of biomembranes to the behaviour of cells. The interaction of cells and the formation of tissues and whole organs are also included. Arguments could be made, to observe the interaction of organisms in ecological systems, for example biofilms or predator-prey relations, in the context of systems biology [Farina and Dennunzio, 2008].

Computational models and simulations represent another very important part of systems biology. A system, independent of its scale, can not be understood by creating a diagram of the interactions. Identifying all parts (genes and proteins) of an organism is like listing all the parts of a semiconductor fabrication plant. These are the first steps to understand the system at hand, but to comprehend and untimely to control the processes that can occur, it is essential to determine how the parts dynamically interact over time. KITANO named four key components to derive this knowledge (the points have been slightly adapted [Kitano, 2002]):

- **System structures**
  including the interaction networks of genes and biochemical pathways as well as their modulation from intra- and inter-cellular signals
- **System dynamics**
  including the behaviour and adoption of the system under varying conditions

- **Control**
  trying to control the system, to get a feeling for its dynamics and limits
- **Design**
  trying to design the system helps to see which components are indispensable and which are fall-back mechanisms.

The emerging phenomena of a cell can be looked at from two different perspectives, that can be compared best to two competing philosophical schools of thought. **Holism** is the principle, that natural systems and their properties should always be viewed as wholes not as a collection of parts. This includes that a system's function can not be viewed solely as a sum of its parts [Auyang, 1999]. Furthermore it states that in science a system could be computationally irreducible, which means it would not be possible to even approximate the system state without a full simulation of all the events occurring. Its complementing school of thought is reductionism. **Reductionism** analyses a complex system by subdivision or reduction to more fundamental parts. The interaction of the parts can then be used to explain the whole phenomenon in its entirety. It has been stated and underpinned that reductionism is in fact unable to explain phenomena crossing different scales of magnitude [Anderson, 1973]. Nevertheless, most systems that have been developed in the course of history are able to use reductionism in a way that makes it possible to approximate the behaviour and emergence of a phenomenon. This state is followed in many areas of sciences, e.g. solid-state physics, cell biology, and medicine. For example the bacterial cell cycle can be explained by a handful of genes and proteins, without knowing all the processes in a cell [McAdams and Shapiro, 2009]. Even though, we will possibly never be able to understand a complex system such as a human from looking at the sum of its parts – we are able to approximate the progression of a disease and alter its course in some cases. By simulating and tinkering with the system it is possible to arrive at a different destination, contrary to the unaltered system. It is not necessary to understand the whole phenomenon, but the parts that are of significant interest for the question asked.

In 2006 the National Science Foundation proposed that systems biology should rise to a grand challenge in the 21st century: to build a mathematical model of the whole cell [Omenn, 2006]. In order to understand a system a complex as a cell the fundamental processes that govern the behaviour of a cell have to be understood first. The processes of diffusion and the chemical reactions in a cell seem to be a good point to start this endeavour.

1.2.2 Diffusion

Classical diffusion

Diffusion is a natural physical process where molecules disperse from an area with a high concentration towards an area with a low concentration. This phenomenon is driven by a random motion which all particles in solutions experience, called Brownian motion [Mehrer and Stolwijk, 2009]. ADOLF FICK posited in 1855 that the flux of matter in a liquid is proportional to the gradient of its concentration with a proportionality factor D [Fick, 1855]. Fick's first law can therefore be described with:

$$J = -D\frac{\partial c}{\partial x} \tag{1.1}$$

where J is the diffusion flux (in $^{mol}/_{m^2 \cdot s}$), D is the diffusion coefficient (in $^{m^2}/_{s}$) and $^{\partial c}/_{\partial x}$ is the concentration gradient (in $^{mol}/_{m^4}$). With the continuity equation of the law of conservation of mass [de Lavoisier, 1801]:

$$\frac{\partial c}{\partial t} = -\frac{\partial J}{\partial x} \tag{1.2}$$

and for constant diffusion coefficients we obtain Fick's second law:

$$\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial x^2} \tag{1.3}$$

The only coefficient for the diffusion is D, therefore also called diffusion coefficient, which can be described by the Stokes-Einstein equation [Einstein, 1905]. The general form of the equation is:

$$D = \mu\, k_B T \tag{1.4}$$

where:

- μ is the mobility or the ratio of the particle's terminal drift velocity to an applied force, $\mu = {v_d}/{F}$,
- k_B is Boltzmann's constant, and
- T is the absolute temperature.

Albeit this equation seems simple, it is very hard to apply it to real world problems. In order to estimate the diffusion, many researchers focused on how to approximate the coefficient with an equation that can be applied to improve and explain experiments. In 1955 WILKE and CHANG delivered correlated diffusion coefficients with the molar volume of the solute, which lead to the function [Wilke and Chang, 1955]:

$$D = 7.4 \times 10^{-8} \frac{(x \cdot M)^{0.5} T}{\eta V^{0.6}} \tag{1.5}$$

where

- T is the temperature of the system (in K),
- M is the molecular weight of the solvent (in $^{g}/_{mol}$),
- V is the molecular volume of the solute (in $^{cm^3}/_{g \cdot mol}$),
- η is the dynamic viscosity of the solution (in $mPas$), and
- x is an association parameter to define the effective molecular weight of the solvent.

The experiments were conducted for different small molecules in multiple solutions. Therefore, the equation is optimized to describe the diffusion coefficient for different solvents and temperatures. The observed temperatures range from 6.6° C (279.75 K) to 40.0° C (313.15 K). The different solvents used are Benzene, Formic acid, Iodine, and Toluene amongst others. The association parameter has been suggested to be $x = 1$ for non-associated solvents and $x = 2.6$ for water [Wilke and Chang, 1955]. Later it has been shown that $x = 2.26$ for water produces a better fit [Sitaraman et al., 1963].

HAYDUK and LAUDIE [Hayduk and Laudie, 1974] developed a correlation for molecules solved in aqueous solutions. This correlation is especially optimized for small molecules in water at room temperature:

$$D = 13.26 \times 10^{-5} \frac{1}{\eta^{1.4} V^{0.589}} \tag{1.6}$$

In the plot pictured in Figure 1.1 both correlations have been evaluated for 86 chemicals (chemicals will be referred to as species, as it is conventional in systems biology) taken from [Hayduk and Laudie, 1974]. Depicted are the molar volume against the diffusion coefficient and the lines of fit for both correlations. Both equations yield very similar results. The advantage of the Wilke correlation is that it can be used for other temperature regimes and solvents, too. Two solutes, Hydrogen (H_2) and Helium (He), don't behave as expected, considering the values of the other species. Both are very light but have a high molar volume of 28.5 $^{cm^3}/_{g \cdot mol}$ and 31.9 $^{cm^3}/_{g \cdot mol}$, respectively. This is a result of the definition of the molar volume. All substances are measured in their natural forms which is gas for the discussed species – but they behave differently when dissolved in a solution.

The following YOUNG correlation [Young et al., 1980] has been suggested especially for proteins in water and under the assumption that most proteins have a partial specific volume between 0.69 and 0.78 $^{cm^3}/_{g}$ [Charlwood, 1957]:

$$D = 8.34 \times 10^{-8} \frac{T}{\eta M^{1/3}} \tag{1.7}$$

Here temperature has to be given in $^{\circ}C$. The data used to calculate the correlation has been taken from multiple sources that measured the diffusivity of proteins in an aqueous

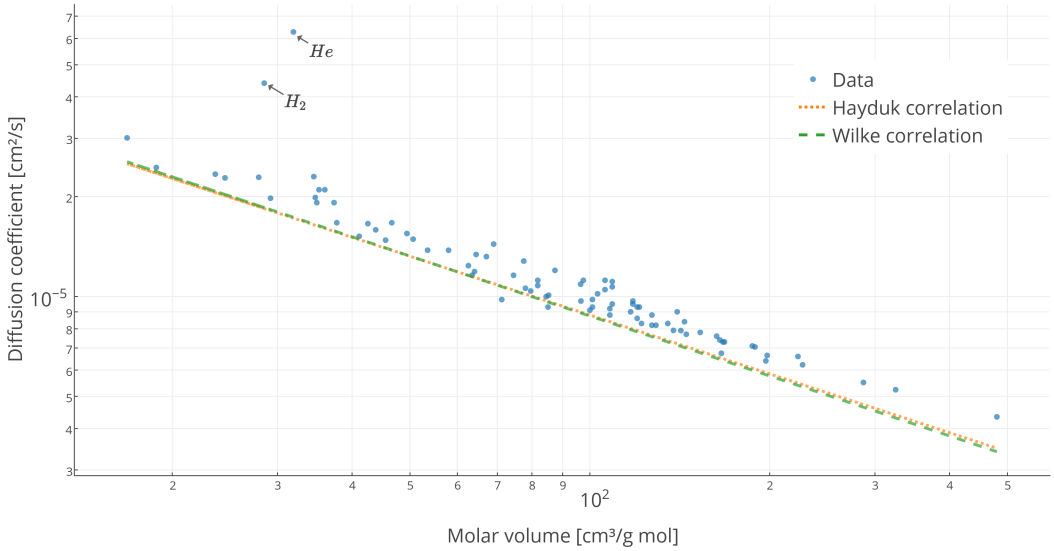


Figure 1.1: Depicted are the molar volumes against the diffusion coefficient of 86 species. The diffusion coefficients and molar volumes have been taken from [Hayduk and Laudie, 1974]. Additionally, both of the described correlations (Equations 1.5 and 1.6) have been calculated for $x = 2.26$, $T = 293K$, $M_{H_2O} = 18.0153\ {}^{g}/_{mol}$, $\eta_{H_2O} = 1\ mPas$, $V_{H_2O} = 18.9\ {}^{cm^3}/_{g \cdot mol}$, where applicable, and are displayed as dashed lines in the plot. The species are listed in Table A.1

solution. Henceforth, the molar weight in the article has been revised by more accurately determined values from the Universal Protein Resource (UniProt) database [UniProt Consortium, 2008]. It is noticeable that the average weight given in the paper was higher than the actual weight of the protein as fetched from the database, but this does not seem to impact the correlation significantly. The data has been plotted in Figure 1.2. Three significant outliers are Collagen, Fibrinogen, and Myosin light chain. These fibrous proteins express diffusion coefficients lower than predicted by the correlation. That has to be expected because their structures are not spheric, as assumed by the modified Stokes-Einstein equation, but elongated.

Those correlations are valid for diffusion, that is unimpaired by any complex interactions. In cellular environments many factors that are hindering the diffusion are omnipresent and therefore the described correlations cannot be used for the application in cells without change.

Cellular diffusion

During the first half of the 20th century the interior of a cell was thought to be viscous gel. This view has evolved to an aqueous but crowded environment [Luby-Phelps, 2000]. Cellular biochemistry is based on studies that observe dilute solution with isolated enzymes and their substrates. However, the number of different small and large molecules inside the cell is very high. The protein content alone is estimated to be between 17%

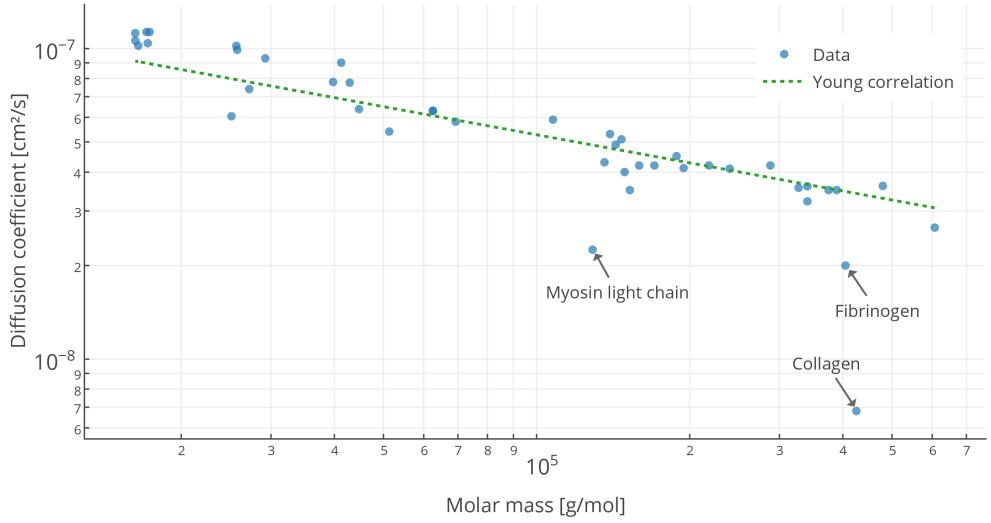


Figure 1.2: Depicted are the molar masses against the diffusion coefficients of 43 proteins. The diffusion coefficients have been taken from [Young et al., 1980]. The molar masses of the proteins have been fetched from the Universal Protein Resource (UniProt) database [UniProt Consortium, 2008] under consideration of the natural subunit structure. Additionally, the described correlation has been calculated for $T = 293K$ and $\eta_{H_2O} = 1$ and is displayed as a dashed line in the plot. The proteins are listed in Table A.2.

and 35% of the cellular weight [Fulton, 1982]. The length of the path that the molecule has to take increases due to obstacles that have to be avoided and electrostatic interactions that influence its movements. The three driving forces behind the mobility of a molecule in the cell have been proposed to be [Kao et al., 1993]:

- the viscosity of the aqueous phase between molecules,
- the solutes affinity to bind to other molecules, and
- the collisions with other macromolecular structures.

Small solutes act as an important indicator to describe the cytoplasmic viscosity. The solute transitional diffusion coefficient (STDF) can be used to approximate the influences of the three described effects [Verkman, 2002] to the overall speed of the diffusion. It has been shown, that the diffusion of 2',7'-Bis-(2-Carboxyethyl)-5-(and-6)-Carboxyfluorescein (BCECF) acid in cellular environments is four times slower than in water (STDF = $D_{cyto}/D_{water} \approx 0.27 \pm 0.01$ at $23°$C) [Kao et al., 1993]. The viscosity of the aqueous phase of the cytoplasmic lumen seems to be similar to that in water as indicated by the rotational correlation times of BCECF and other small solutes, that were only 10 – 30% slower in cytoplasm. The binding of the observed solute to other molecules accounts for approximately 20% of the decreased speed. The main hurdle for a fast diffusion is the collision with other cellular structures and molecules. This has been proven by bloating cells with saline solution and vice versa shrinking them by wa-

ter scarcity [Luby-Phelps et al., 1993]. Figure 1.3 shows the molar mass of a molecule against its STDF (D_{cyto}/D_{water}). The proportion of which a molecule is slowed seems to be constant up to a mass of 100 kDa. But there are some exceptions. Again fibrous and long molecules (see Dextran and DNA in Figure 1.3) are significantly slowed. The diffusion of glycolytic enzymes has also been studied and found to be significantly impaired in some cases [Pagliaro and Taylor, 1992]. There is evidence that the diffusion of some glycolytic enzymes might by regulated by cell's metabolic state [Verkman, 2002]. The metabolic state, or the concentrations of chemical key species, can be described with different kinds of chemical kinetics.

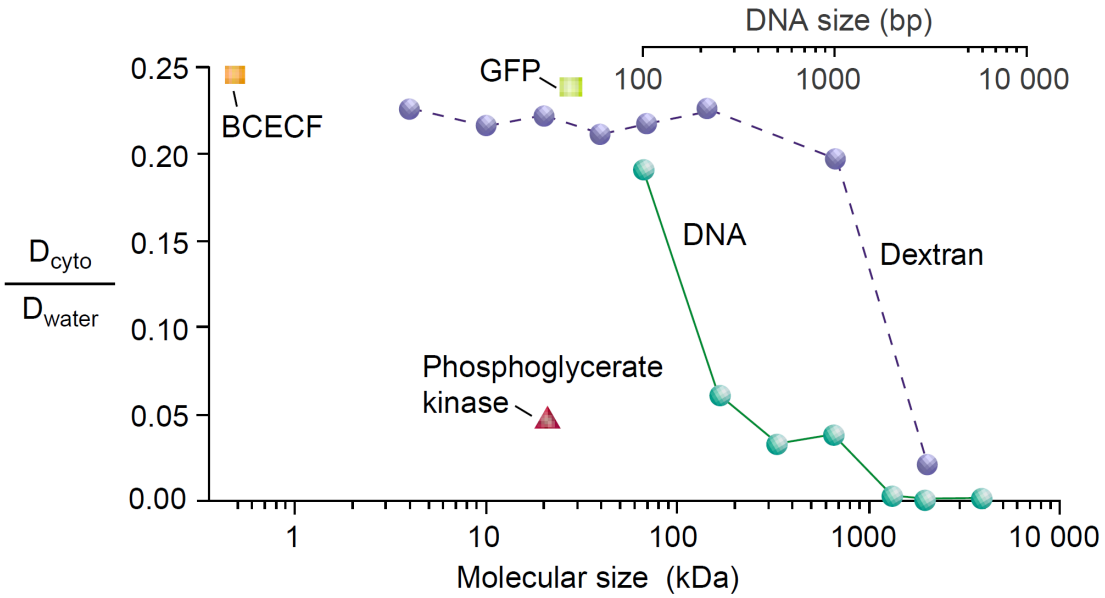


Figure 1.3: Depicted are the molar masses against the STDF of multiple molecules. The diagram has been adopted from [Verkman, 2002].

1.2.3 Chemical kinetics

In 1850 the German scientist WILHELMY was one of the first to describe that the velocity of a chemical reaction follows certain laws [Wilhelmy, 1850]. He observed that the concentrations of the substrates sucrose and acids influence the reaction's rate directly. To interpret his experimental result he designed a differential equation and integrated it. With this fundamentals WAAGE and GULDBERG started to explore the field of chemical kinetics by postulating the law of mass action that explicitly states that the chemical affinity is proportional to the masses (or concentrations) of two substrates A and B [Guldberg and Waage, 1864].

Chemical kinetics classify chemical reactions according to their molecularity or their order. Molecularity defines the number of molecules that are altered by a reaction. A reaction $A \rightarrow P$ is unimolecular and $A + B \rightarrow P$ is bimolecular. Other one step reactions of higher molecularity are extremely rare. The classification by reaction order depends on the number of concentrations that influence the rate of the overall reaction. Therefore, first-order reactions only depend on one substrate, second order reactions on two, and so on. There are also some reactions that have been observed to be of zero order, if all reactants are available at very large excess. [Cornish-Bowden, 2013]

In the general case of a reaction of two substrates denoted as:

$$aA + bB \rightarrow pP + qQ + \cdots$$

the rate of reaction can be described by [McNaught and Wilkinson, 1997]:

$$v = -\frac{1}{h(A)}\frac{dc(A)}{dt} = -\frac{1}{h(B)}\frac{dc(B)}{dt} = \frac{1}{h(P)}\frac{dc(P)}{dt} = \frac{1}{h(Q)}\frac{dc(Q)}{dt} = \cdots \tag{1.8}$$

where:

- v is the reaction rate (in $^{mol}/_{l \cdot s}$),
- $h(S)$ is the stoichiometric coefficients of a species S,
- $c(S)$ is the concentration or amount of a species S, often given (in $^{mol}/_{l}$), and
- t is the time.

First-order kinetics

The reaction rate of a first-order reaction $A \rightarrow P$ can be described by:

$$v = \frac{dc(P)}{dt} = -\frac{dc(A)}{dt} = kc(A) = k \cdot (c_0(A) - P) \tag{1.9}$$

where k is the rate coefficient.

To find a solution the rate law can be integrated by separating the two variables $c(P)$ and t:

$$\int \frac{dc(p)}{c_0(A) - c(P)} = \int k dt \tag{1.10}$$

This leads to the integrated first order rate law [Cornish-Bowden, 2013], which is usually written in the form of an exponential decay equation:

$$c_t(P) = c_0(A)(1 - e^{-k \cdot t}) \tag{1.11}$$

This solution makes it possible to determine the concentration of a product at any time t, using the starting concentration of the substrate $c_0(A)$ and the rate constant k. The generalized equation for different stoichiometric coefficients is:

$$c_t(S) = \begin{cases} \frac{h(S)}{h(A)} \cdot c_0(A) \cdot (1 - e^{-k \cdot h(A) \cdot t}) & \text{if S is a product,} \\ \frac{h(S)}{h(A)} \cdot c_0(A) \cdot (e^{-k \cdot h(A) \cdot t}) & \text{if S is a substrate.} \end{cases} \tag{1.12}$$

Second-order kinetics

The reaction rate of a second-order reaction $A + B \rightarrow P + Q + \cdots$ can be described by

$$v = -\frac{dc(A)}{dt} = -\frac{dc(B)}{dt} = \frac{dc(P)}{dt} = \frac{dc(Q)}{dt} = k \cdot c(A)^{\alpha} \cdot c(B)^{\beta} \tag{1.13}$$

where k is the rate coefficient, α and β are partial reaction orders. The exponents α and β are generally experimentally determined small, positive integers, but may also be zero, fractional or negative [Capellos and Bielski, 1980]. They can be associated to, but are not necessarily the stoichiometric coefficients.

In chemical kinetics the half-life time of a reaction describes the time needed for half of the reactant to be depleted. The half-life of a second order reaction can be calculated with [Capellos and Bielski, 1980]:

$$\tau_{1/2} = \frac{1}{k \cdot c_0(A)} \tag{1.14}$$

Basically there are two different cases where second-order rate kinetics are applied. In the first case, two of the same reactants are combined: $A + A \rightarrow P = 2A \rightarrow P$. The reaction rate can then be given by $v = kc(A)^2$. In the other case two different substrates react to a product $A + B \rightarrow P$ and the resulting rate of reaction is $v = k \cdot c(A)^{\alpha} \cdot c(B)^{\beta}$.

Reversible Reactions

Chatelier's principle states that any change to a system at its equilibrium results in an opposing reaction in the corresponding system [Le Chatelier and Boudouard, 1898].

The knowledge of equilibrium constants is essential for understanding many chemical systems. For example, the dissociation and association of protein ligand binding is a result of equilibrium reactions [Lodish et al., 2000]. A chemical reaction is at chemical equilibrium when all reactants and products are present in concentrations which have no further tendency to change with time [Atkins and De Paula, 1998]:

$$A \underset{k_b}{\overset{k_f}{\rightleftharpoons}} B$$

This means that the reaction exists in an equilibrium, when the forwards reaction rate k_f and backwards reaction rate k_b are equal.

$$v = -\frac{dc(A)}{dt} = \frac{dc(B)}{dt} = k_f \cdot c(A) - k_b \cdot c(B) \tag{1.15}$$

The integrated reversible reaction rate law [Cornish-Bowden, 2013] can be used to calculate the concentration of the product at time t:

$$c_t(P) = \frac{k_f \cdot c_0(A) \cdot (1 - e^{-(k_f + k_b) \cdot t})}{k_f + k_b} \tag{1.16}$$

Interestingly, the addition of a catalyst does not influence the equilibrium constant, it will only affect the speed at which the equilibrium is reached, because both reaction rates k_f and k_b are influenced evenly. An equilibrium constant can be derived from the reaction quotient when the reaction has reached the equilibrium and can be described in two ways. One is to define it as a thermodynamic constant that uses the thermodynamic activity $a(S)$ with:

$$k_{eq} = \frac{a(B)^{h(B)}}{a(A)^{h(A)}} \tag{1.17}$$

Another way uses the concentration of the involved species:

$$k_{eq} = \frac{c(B)^{h(B)}}{c(A)^{h(A)}} \tag{1.18}$$

Both resulting constants are dimensionless. The constant k_{eq} is often determined experimentally [Eggleston and Hems, 1952, Collett et al., 2015].

Michaelis-Menten kinetics

The kinetics of enzyme-catalysed reactions were first extensively studied and analysed by EDUARD BUCHNER. He showed that cell free extract from yeast could catalyse alcoholic fermentation [Buchner, 1897]. Most of those early studies dealt with fermentation, today known as the hydrolysis of sucrose:

$$\text{sucrose} + \text{water} \rightarrow \text{glucose} + \text{fructose}$$

The rate law was found to be proportional to the amount of enzyme and decreased when the substrate was consumed. Additionally, and unlikely to "regular" reactions, there was an optimum temperature. Above this optimal temperature, the rate quickly desecrated towards zero. More astoundingly, the catalyst invertase was 100,000 times heavier than its substrate sucrose and not consumed by the reaction [O'Sullivan and Tompson, 1890]. From a kinetic standpoint, it was found that the rates deviated from second order kinetics and it was suggested that the involvement of an enzyme-substrate complex in the mechanism provided a threshold for the maximal reaction rate [Brown, 1892].

The mechanism of enzyme reaction as it is accepted today has been developed by [Henri, 1903]:

$$E + A \underset{k_b}{\overset{k_f}{\rightleftharpoons}} EA \overset{k_2}{\rightarrow} E + P \tag{1.19}$$

The first equilibrium reaction can be given as an equilibrium constant for substrate dissociation $k_{eq} = {}^{c(E)\cdot c(A)}/_{c(EA)}$. The free concentrations of E and A can only be measured at the beginning of the experiment and are given as $c_0(E)$ and $c_0(A)$. Further, the initial concentrations are then determined by $c_0(A) = c(A) + c(EA)$ and $c_0(E) = c(E) + c(EA)$. The concentration of A is in the majority of reactions a lot larger than the concentration of E: $c(A) \gg c(E)$. Subsequently, $c(EA)$ is much smaller than $c(A)$ and therefore $c_0(A) = c(A)$ as a good approximation. Now $c(EA)$ can be derived by:

$$c(EA) = \frac{c_0(E)}{(\frac{k_{eq}}{c_0(A)}) + 1} = \frac{c_0(E)}{(\frac{k_{eq}}{c(A)}) + 1} \tag{1.20}$$

The second partial reaction $EA \rightarrow E + P$ is a simple first-order reaction, with a rate constant k_2:

$$v = k_2 \cdot c(EA) = \frac{k_2 \cdot c_0(E)}{(\frac{k_{eq}}{c(A)}) + 1} = \frac{k_2 \cdot c_0(E) \cdot c(A)}{k_{eq} + c(A)} \tag{1.21}$$

Those first steps to understand enzyme kinetics were made by MICHAELIS and MENTEN [Michaelis and Menten, 1913], which the final equation will be named after. Nevertheless, a big part of the generalisation of the equation has been achieved by what is called the "BRIGGS-HALDANE treatment" [Briggs and Haldane, 1925]. They argued that a steady state would be reached where the concentration of the enzyme-substrate complex was constant: ${}^{dc(EA)}/_{dt} = 0$. Finally the following equation was derived:

$$v = \frac{k_2 \cdot c_0(E) \cdot c(A)}{\frac{k_b + k_2}{k_f} + c(A)} = \frac{k_{cat} \cdot c_0(E) \cdot c(A)}{k_m + c(A)} \tag{1.22}$$

The catalytic constant k_{cat}, also called turn-over number, is an enzyme specific constant that describes the number of catalytic cycles the enzyme can undergo in a time step (e.g. a second). The so-called *Michaelis* constant $k_m = {}^{(k_b + k_2)}/_{k_f}$ symbolizes the affinity

of the enzyme-substrate complex. Because in experiments very high concentrations of both substrate and enzymes are used, the equation becomes dominated by the concentration of the substrate and therefore the often used form of the equation is given by:

$$v = \frac{V_{max} \cdot c(A)}{k_m + c(A)} \tag{1.23}$$

where all enzymes are saturated and a maximal velocity V_{max} can be denoted [Cornish-Bowden, 2013].

1.2.4 Mathematical modelling

Mathematical modelling means describing a real world problem in a mathematical way, so that it becomes solvable with mathematical methods. The accuracy of the description should be limited, so that it does not get unnecessarily complex [Mattheij et al., 2005]. Sadly, there is no universal concept that always leads to a satisfying description of a problem. Mathematical modelling is therefore not as concerned with finding a solution, but rather phrasing the appropriate question. Subsequently, this question can be solved with the means of mathematics. Starting point is therefore a real-world problem that has to be transformed into a mathematical problem. This depiction of the problem will then be analysed and eventually solved with simulations and/or computer algebra systems. Afterwards, the mathematical solution will be interpreted in context of its real-world implications. Finally, the real-world solution can be validated with the starting problem. A schematic description of this process is given in Figure 1.4.

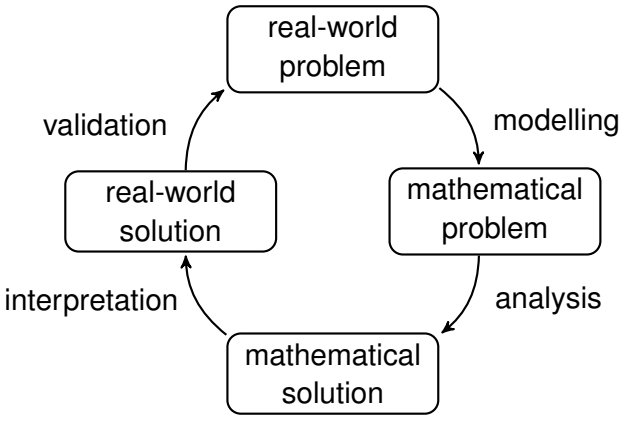


Figure 1.4: A schematic description of the modelling process.

The toolbox of mathematics is vast. Consequently, the usage of the right implementation is yet another question that has to be addressed. The tools that exist today are very powerful, but – as in everyday life – one can use a hammer to steady a screw, however a screwdriver would do the job more elegantly.

In the late 19th century HEINRICH HERTZ postulated three principles for mathematical models [Hertz, 1863]. He also highlights that models are not unique, it is therefore possible that more than one correct model may exist. Those models can omit multiple traits that can be preferable over one another, e.g. one model could be easier to solve whereby another model is more scalable. The three essential criteria are:

- **validity**
  The validity or "correctness" of a model always results from compliance with the phenomena of the real-world problem.
- **legitimacy**
  A model is legitimate or logic, as far as it is unambiguous and does not contain

inconsistencies. This criterion can only be validated by mathematics and if the question asked is precise enough.

- **practicability**
  A model is practicable, as far as it only contains elements that are beneficial to the solution. Amongst two competing hypotheses that predict equally well, the simpler solution should be preferred (a principle called Occam's razor).

Mathematical modelling can be used to construct different mathematical systems. Those systems can be classified by certain characteristics. A difference can be made between **explicit** and **implicit** systems. If the state of a system at a later time wa strictly dependent on the system at its current state it's called explicit. On the other hand the solution to an implicit system involves both the current state of a system and the later one. This can be described via the definition of a system $Y(t)$ in its current state and $Y(t+\Delta t)$ – its state at a later time. An explicit system calculates the next state by $Y(t+\Delta t) = F(Y(t))$ while an implicit system has to solve the equation $G(Y(t), Y(t+\Delta t)) = 0$. Further, a model can be **deterministic** or **probabilistic**. A deterministic model arrives always at the same solution given the same starting conditions. Alternatively, a probabilistic model describes a system with events that occur with a certain possibility, therefore it is possible to arrive at different final states even with the same starting conditions. [Brin and Stuck, 2002]

While **static** models calculate the state of a system in an equilibrium and are therefore time independent, **dynamic** models describe time dependent changes from a set of starting variables.

Dynamical systems

Dynamical systems help to study the long-term behaviour of evolving systems [Brin and Stuck, 2002]. The notion of a dynamical system originates in Newtonian mechanics. Those, so called classical mechanics, are laws that describe the motion of bodies over time, starting from an initial state, where all relevant variables are known. To determine the state of a variable at a specific time t_x, the state of all bodies has to be computed at a small step at a time point. This naive approach resulted in the large mathematical field of calculus and ultimately in differential calculus and integral calculus. Infinitesimal small changes in space and time were essential to determine the laws that describe a system over large periods of time. Besides those so-called continuous dynamical systems, there are discrete dynamical systems that do not assume infinitesimal small time steps, but rather in jumps or epochs.

Definition 1.1 (Discrete dynamic system) Let a discrete dynamic system (DDS) be a pair (M,T), where

- M is a non empty set (the state space),
- $T : M \mapsto M$ is a function that maps M to itself (law of motion).

The set M commonly contains more complex structures that make up the state space of the dynamic system and the function T describes the "movement" of the system to the next state $m \in M$. If m_0 is the first state the following states are defined by $m_i = Tm_{i-1}$. The state m_i is then the state at time $i \in \mathbb{N}$. The term "discrete" describes that the time is split into intervals $i = \{0,1,2,\ldots\}$ [Krause and Nesemann, 2012].

Definition 1.2 (Fixed points) Let $D = (M,T)$ be a discrete dynamic system, then $m^* \in M$ are states, such that

$$Tm^* = m^*$$

So-called fixed points are of major importance when analysing discrete systems. Fixed points can either be attractors or repellents. Attractors are fixed points that describe equilibria dynamic systems strive towards, if i approaches infinity. In case the system is at a repellent fixed point and experiences a small change, the trajectory of the system is rejected from that point.

Discrete time

One amongst many similar definitions for time reads as: "A quantity used to specify the order in which events occurred and measure the amount by which one even preceded or followed another" [Duffett-Smith, 1992]. Especially in mathematics time can be perceived as continuous ($T \in \mathbb{R}$) or discrete ($T \in \mathbb{Z}$), but, when developing a model one of the two has to be chosen eventually. However, often this decision is not made conscious but because of the underlying fundamentals, which demand the one or the other. Arguments that are often used in favour of continuous time are [Érdi and Tóth, 1989]:

- time is continuous in reality
- solutions of continuous systems can be calculated quick, if the corresponding differential equation has been solved
- in the past a lot of models using continuous time have successfully been developed (e.g standard quantum mechanics and general relativity).

On the other hand discrete time can be favoured because:

- in reality time is discrete
  Time is a continuous quantity in standard quantum mechanics, nevertheless it

has been suggested multiple times a discrete time can be just as suitable [Lévi, 1927, Planck, 1943, Margenau, 1950]. The Planck time t_p is the minimal unit of time in the system of natural units. It describes the time for light to travel a distance of one Planck length in vacuum and can therefore be described by $t_p \equiv \sqrt{\hbar G/c^5} \approx 5.39106 \times 10^{-44}$ s.

- calculations with discrete-time models are simple
- computer can only calculate discrete time steps
- experiments are measured at discrete points.

This work uses a discrete time adaptation of equations that have originally been designed for continuous solutions. Discretisation can be used to replace infinitesimal small time steps with steps of a certain duration. This results in a recurrence relation. The recurrence relation defines a sequence of values once an initial term is given, each further term is defined as a function of the preceding terms [Krause and Nesemann, 2012].

Remark (Discrete time) The current time step is denoted by t, the next time step with $t+1$, the previous with $t-1$, and every other analogously. A time segment $\Delta t_i = t_i - t_{i-1}$ describes the time between two time steps.

1.2.5 Graphs and automata

Graphs

A graph is a set of objects, called nodes or vertices. Those nodes are connected by links, also called edges.

Definition 1.3 (Undirected graph) An undirected graph is a triple $G = (V, E, \phi)$, where

- V is a finite set (the vertices or nodes),
- E is a finite set (the edges), and
- $\phi : E \mapsto V \cup V$ is a partial function (the incidence relation).

The incidence relation $\phi(e)$ maps an edge $e \in E$ to a number of nodes $v \in V$. Two edges $e, e' \in E$ are called **parallel** if $\phi(e) = \phi(e')$. An edge is called a **loop** if $\phi(e) = \{v\}$. Two nodes $v_i, v_j \in V$ are called **neighbours** if there is an incidence relation such that $\phi(e) = \{v_i, v_j\}$. If two nodes are neighbours they are **adjacent** to each other. The **degree** $deg(v)$ of a node is determined by the number of its neighbours. Additionally, a graph can be **weighted** with a function $\omega : E \mapsto \mathbb{R}$ that maps a real number w to an edge e [West, 2001].

A graph is called **simple** in case it contains neither parallel edges, nor loops. The definition can then be simplified to $G = (V, E)$ with E being a set of subsets of V with

two elements:

$$E \subseteq \big\{\{v_i, v_j\} \subseteq V | v_i, v_j \in V, v_i \neq v_j\big\} \tag{1.24}$$

A graph can be depicted with a graphical representation as shown in Figure 1.5.

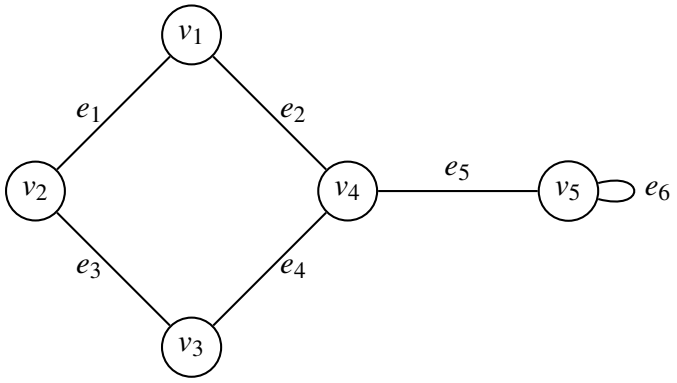


Figure 1.5: The graph contains five nodes $v_i \in V$ represented as circles and six edges $e_j \in E$ connecting the nodes.

Additionally, a graph can be represented on a computational level with edge lists, adjacency matrices or adjacency lists, amongst others. To represent a graph G with $|E|$ edges with an **edge list**, the definition of ϕ in Definition 1.3 is applied. In case of the graph depicted in Figure 1.5 the edge list representation of all elements of ϕ is as follows:

$$\big\{\{1,2\},\{1,4\},\{2,3\},\{3,4\},\{4,5\},\{5,5\}\big\}.$$

In case edges are weighted, a third element can be added to this representation. Since each element of the list contains only two or three numbers, the total space requirements for an edge list is $\Theta(|E|)$. Retrieval of a specific edge with a linear search has a time complexity of $\mathrm{O}\big(|E|\big)$. The edges can be organized to achieve a better time complexity of $\mathrm{O}\big(1\big)$, at cost of an total space increase to $\Theta(|V|^2)$. For a graph with $|V|$ vertices, an **adjacency matrix** is a $|V| \times |V|$ matrix consisting of elements $m_{i,j} \in \{0,1\}$, where the entry in row i and column j is 1 if, and only if, the edge (i, j) exists in the graph. The weight can be incorporated by putting the weight $w \in \mathbb{R}$ of an edge (i, j) in row i and column j of the adjacency matrix. For an undirected graph the matrix is symmetric. Again for the graph in Figure 1.5 the adjacency matrix is as follows:

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}.$$

Adjacency lists are a combination of both approaches. For each vertex a list with

adjacent edges is stored. The time complexity to get to the neighbours of vertex v_i is constant $\mathrm{O}(1)$. To detect, if an edge (i, j) was present in a graph, a search algorithm has to retrieve whether j is in the neighbour list of i and has therefore a time complexity of $\mathrm{O}(deg(v_i))$. Each edge appears twice in the adjacency list, which leads to a space requirement of $\mathrm{O}(2|E|)$. If no loops were in the graph, the space requirement can be tightly described by $\Theta(2|E|)$. For example, the graph in Figure 1.5 can be represented as:

$$1 \mapsto \{2,4\}, 2 \mapsto \{1,3\}, 3 \mapsto \{2,4\}, 4 \mapsto \{1,3,5\}, 5 \mapsto \{4,5\}.$$

In summary, adjacency matrices are superior for representations of small graphs because space requirements are high but searches are very fast. Edge lists are suitable for graphs where finding a specific edge is not a critical problem, it provides a less redundant form to store information of a graph. Adjacency lists compromise a fast search strategy and moderate space requirements.

Table 1.1: Growth of space requirements and the time complexity of a linear search of an edge in the graph. Compared are the described graph representations.

	edge list	**adjacency matrix**	**adjacency list**
space requirements	$\Theta(\|E\|)$	$\Theta(\|V\|^2)$	$\mathrm{O}(2\|E\|)$
linear search of (v_i, v_j)	$\mathrm{O}(\|E\|)$	$\mathrm{O}(1)$	$\mathrm{O}(deg(v_i))$
remark	slow to search edges and neighbours	slow to add vertices, because array has to be resized	slow to delete vertices, because the whole list has to be checked

Deterministic finite automata

In theoretical computer science, a deterministic finite automaton (DFA) accepts or rejects a finite string of symbols [Hopcroft et al., 2002].

Definition 1.4 (Deterministic finite automaton) A DFA is a quintuple $A = (Q, \Sigma, \delta, q_0, F)$, where:

- Q is a finite set (the states),
- Σ is a finite set (the input alphabet),
- $\delta : Q \times \Sigma \mapsto Q$ is a partial function (the transition function),
- $q_0 \in Q$ is the initial state, and
- F is the set of final states.

Beginning at the starting state q_0 a word $w \in \Sigma^*$ leads to another state. The transitions are determined by the transition function which maps from a state $q_i \in Q$ to another state

$q_j \in Q$ given by a letter $a \in \Sigma$. An automata accepts a word w in case the transition leads to a final state $q_f \in F$ [Dohmen, 2013].

A graphical representation of a DFA is depicted in Figure 1.6. The formal definition of this automaton is as follows: $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0,1\}$, $\delta(q_0,0) = q_1$, $\delta(q_1,1) = q_2$, $\delta(q_2,0) = q_2$, $\delta(q_2,1) = q_3$, and $F = \{q_3\}$.

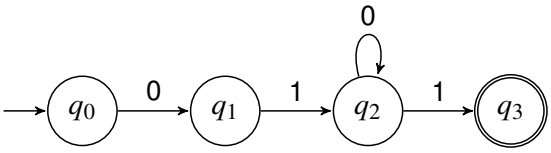


Figure 1.6: The DFA contains four states $q_i \in Q$ represented as circles and four transitions. The initial state q_0 is highlighted with an initial arrow coming from outside of the automaton and the final state q_3 is emphasized with a doubled border.

Cellular automata

Discovered by STANISLAW ULAM and JOHN VON NEUMANN in the 1940s, cellular automata are a simple way to describe complex interactions [Adami, 1998]. Technically, a cellular automaton is a collection of discrete automata that can interact with each other by reacting to the state changes of their neighbours, given a simple set of rules. One amongst a set of possible definitions is based on the definition of DFA (see Definition 1.4) and will be applied in this work [Thomas, 2002].

Definition 1.5 (One-dimensional cellular automaton) Let Z be a sequence of cells, where every point $z \in Z$ is an automaton $A = (Q, q_0, \delta, q_+)$, where:

- Q is a finite set (the states),
- $q_0 \in Q$ is the idle state,
- $q_+ \in Q$ is the accepted state,
- $\delta : Q^3 \mapsto Q$ is a total function (the transition function).

The transition function $\delta(p,q,r) = q_i$ is defined by the state p of the preceding automaton (at z_{i-1}), q is the state of this automaton (at z_i) and r is the state of the subsequent automaton (at z_{i+1}). Often one-dimensional cellular automata are visualized by bands of rectangles (see Figure 1.7). The next state of the automaton is subsequently appended to the bottom of the previous state.

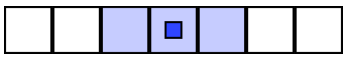


Figure 1.7: Example of one-dimensional cellular automaton. The neighbouring cells of the central automaton are indicated in light blue.

While DFA process each letter $a \in \Sigma$ of a word $w \in \Sigma^*$ individually, a cellular automaton acts at every position simultaneously. The **configuration** C of a cellular automaton A is a sequence of states with $p \in Q$ - given by a word w:

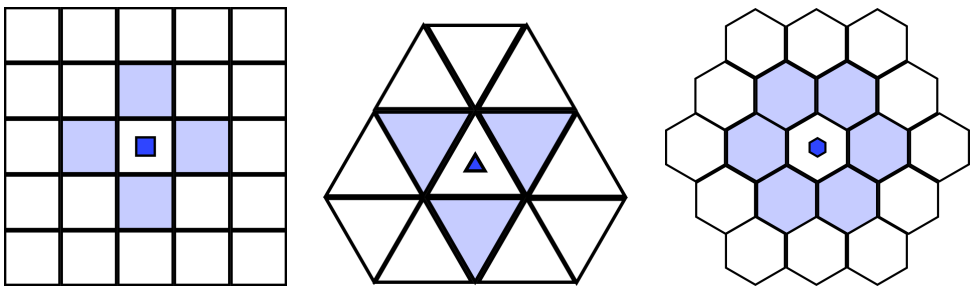
$$C = \ldots, p_{-2}, p_{-1}, p, p_1, p_2, \ldots$$

To transition from a configuration C_t to the next configuration C_{t+1} the function δ is applied. A word is accepted if any automata in Z reaches state q_+. The cellular automaton can also be transformed to a non-finite automaton by omitting the final state q_+. This is done in most applications where cellular automata are used. The rule δ can be given by a table that describes the result of every possible input configuration of states (see Table 1.2). For one-dimensional automata the rules can also be referenced by a so-called WOLFRAM code. The code is generated by translating the result of each possible outcome of the transition function in binary and subsequently transforming it to decimal. For example the rule presented in Table 1.2 is rule $01011010_2 = 90_{10}$ in *Wolfram* code.

Table 1.2: The table includes a rule set for a cellular automaton. Given a sequence of neighbours p, q, r the result of the transition function is shown in the same column.

p, q, r	111	110	101	100	011	101	001	000
$\delta(p, q, r) = q_{t+1}$	0	1	0	1	1	0	1	0

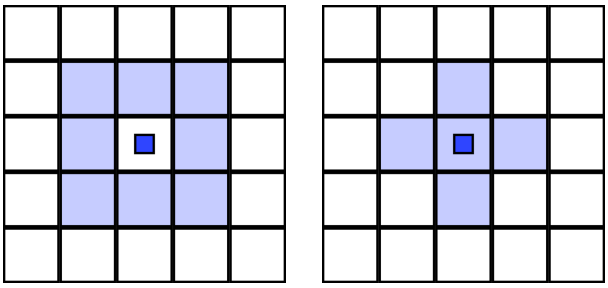
Multidimensional automata can be defined by adjusting the **dimensionality** of Z to Z^n where $n \in \mathbb{N}$. The neighbourhood of automata (p, q, r in the simplest form) can also be altered. If $n = 1$, a neighbourhood relation N can given by a radius r, where r describes the amount of cells to the left and right of the automaton, that are included in the calculation of the transition function. The whole neighbourhood consists of $|N| = 2r + 1$ cells. In two-dimensional automata ($n = 2$) the cells are represented by simple regular shapes. Only three congruent shapes are suitable to build a regular **tiling**. Those are equilateral triangles, squares, and regular hexagons (see Figure 1.8). Other non-regular tilings are possible, but rarely used. If not indicated otherwise, cells that share an edge are considered to be neighbours.



(a) rectangular tiling (b) triangular tiling (c) hexagonal tiling

Figure 1.8: Examples of two-dimensional cellular automata with different tilings. The neighbouring cells of each central automaton are indicated in light blue.

Special neighbourhood relation can also be defined. For example the MOORE neighbourhood includes all cells that share an edge or corner with the cell, but not the cell itself. The VON NEUMANN neighbourhood includes all cells that share an edge and the cell itself (see Figure 1.9).



(a) MOORE neighbourhood (b) VON NEUMANN neighbourhood

Figure 1.9: Example of two-dimensional cellular automata with different neighbourhood relations. The neighbourhood of each central automaton is indicated by light blue.

The last important consideration when handing systems with cellular automata is the **boundary condition**. The cellular space of a cellular automaton is limited, therefore some automata will touch the boundaries of the system. Because the rules are often not applicable for cases, where not all of the neighbours are present, they have to be treated separately. There are mainly two solutions to this circumstance. The first solution is to treat the boundary as uniform. When an automaton touches the boundary, the non-existing cells are set to be in the idle state q_0. The other solution is to treat the boundaries as periodic. For $n = 1$ the cells form a ring and systems with $n = 2$ form a torus. Intuitively, the boundary influences the system directly.

Cellular automata have been used in a variety of applications and interestingly a variety of natural systems behave like cellular automata. The shell of *Cymbiola innexa* expresses a pattern that can be recreated with a simple one-dimensional cellular automaton with rule 30 [Jamtveit and Meakin, 1999]. The growth of crystals and patterns of snowflakes can be modelled by a simple two-dimensional cellular automaton [Gaylord and Nishidate, 2013]. Complex systems in biology (Predator-prey dynamics [Farina and Dennunzio, 2008]) and chemistry (BELOUSOV-ZHABOTINSKY reaction [Madore and Freedman, 1983]) have also been developed with systems of cellular automata.

2 Modelling

2.1 Construction of graph automata

Based on both definitions 1.3 and 1.4 the structure of a graph automaton has been constructed. The resulting system is explicit, meaning the next state of the system is solely determined by its current state. Furthermore, because no functions that involve probability are used, the system is deterministic.

Definition 2.1 (Graph automaton) Let a graph automaton (GA) be a quadruple $GA = (V, E, Q, \phi)$, where:

- V is a finite set of graph automaton cells (see Def. 2.2) (the nodes),
- E is a finite set (the edges),
- Q is a finite set (the states), and
- $\phi : E \mapsto V \cup V$ is a partial function (the incidence relation).

The set V contains the vertices v, that are connected with edges $e \in E$, analogous to their definition in graphs. The set of states is the set of possible conditions a graph automaton cell can adopt. The function ϕ maps a pair of two nodes (v_1, v_j) to each edge e, that are then called neighbours.

Definition 2.2 (Graph automaton cell) Let a graph automaton cell (GAC) be a triple $GAC = (N_s, \delta, q)$, where

- N_s is a multiset (the neighbour states),
- $\delta : N_s^* \mapsto Q_{GA}$ is a total function (the transition function)
- $q \in Q_{GA}$ is the current state.

While DFAs operate with an input alphabet Σ, a GAC operates depending on their neighbours in the graph. The multiset N_s, called **neighbour states**, can be described as a double $N_s = (A, m)$ with A being a finite set and $m : A \mapsto \mathbb{N}$ a function. The function $m(a)$ maps the number of occurrences to every element $a \in A$. If $A \subset Q_{GA}$, then the multiplicity m of $Q_{GA} \setminus A$ is 0. The **neighbourhood** is defined by the states q_{v_n} of the GAC v_n that are adjacent to this GAC v_c (see Figure 2.1):

$$A_{N_s} = \left\{ q_{v_n} \in Q_{GA} | v_c \in \phi(e) \forall e \in GA \right\} \tag{2.1}$$

The cardinality of the multiset is the sum of all multiplicity functions:

$$|A| = \sum_{a \in A} m(a)$$

The transition from one state q_t to another state q_{t+1} is defined by the **transition function** δ and can be deterministic or stochastic. All transitions in a GA occur in the same time segment Δt_i. The domain N^* of δ consists of possible multiplicities of the neighbours of a cell and maps to the new state q_{t+1}, e.g. see Figure 2.2:

$$\delta(N_s) = \begin{cases} q_1 & \text{if } m(q_1) > \frac{|A|}{2}, \\ q_2 & \text{if } m(q_2) > \frac{|A|}{2}, \\ q_3 & \text{else.} \end{cases} \tag{2.2}$$

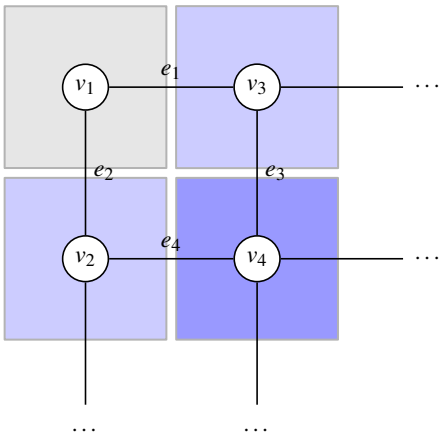


Figure 2.1: The section of a GA contains four GAC $v_i \in V$ represented as circles and four neighbour relations $e_i \in E$. The current cell v_c is highlighted in a dark blue background while its neighbours are coloured in light blue.

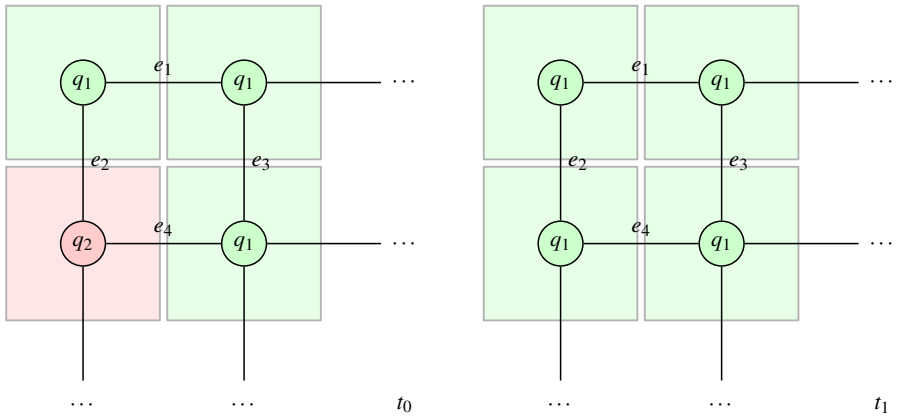


Figure 2.2: A section of a GA is depicted. As the transition occurs the state rule from Equation 2.2 is applied. If more neighbours of a node v_n were in state q_1, its next state will be q_1.

The different manifestations of the transition functions δ are also called **state rules**. After every transition states q_{t+1} have been computed with a given rule, the current states q_t are set to their respective new states q_{t+1}.

With this definition all systems of cellular automata, presented in the introduction, can be recreated. The different neighbourhoods and boundary conditions can be created by adding, respectively removing edges that connect nodes.

In order to handle concentrations of species and ultimately simulate diffusion and kinetics this definition needs to be expanded. Therefore extended graph automaton cells are used to enrich the concept of GA.

Definition 2.3 (Concentration set) A concentration set (CS) is a double $C = (S, c)$, where:

- S is a set (the species), and
- $c : S \mapsto \mathbb{R}$ is a total function (the concentrations).

Each species $s \in S$ represents a chemical compound with its properties. The function $c(s) \in \mathbb{R}$ maps a value to the species s that represents its concentration.

Definition 2.4 (Concentration collection function) Let $\sigma : C^* \times s \mapsto N_c$ be a function, where the domain consists of C^*, a set of multiple concentration sets $C^* = \{C_1, C_2, \ldots, C_m\}$ with $m \in \mathbb{N}$, and s a species, that maps to a multiset $N_c = \{c_1, c_2, \ldots, c_m\}$ $c \in \mathbb{R}$:

$$\sigma(C^*, s) = \{c_1(s), c_2(s), \ldots, c_m(s)\} \tag{2.3}$$

The concentration collection function (CCF) collects the concentration values of a specific species from a set of CS. Subsequently, the function is used to collect the concentrations from the neighbouring GAC. This definition is applied to a graph automaton cell to allow for distribution of real values between the cells.

Definition 2.5 (Extended graph automaton cell) An extended graph automaton cell (EGAC) is a sextuple $EGAC = (N_s, C^N, \delta, \omega, q, o)$, where

- N_S is a multiset (the neighbour states),
- C^N is a set (the neighbour CS (see def. 2.3)),
- $\delta : N_S^* \mapsto Q_{GA}$ is a total function (the transition function),
- $\omega : C^N \times s \mapsto C$ is a total function (the value function),
- $q \in Q_{GA}$ is the current state, and
- C is the current CS.

The definition of GAC (Def. 2.2) also applies to the multiset N_S, the transition function δ

and the current state q. New are the value function ω, the current CS C, as well as the neighbour CS C^N.

The neighbour values $C^N = \{C_{v_1}, C_{v_2}, \ldots, C_{v_n}\}$ are defined similar to N_S in the Definition 2.2.:

$$C^N = \{C_{v_n} | v_c \in \phi(e) \forall e \in GA\} \tag{2.4}$$

Each cell that shares an edge with this cell v_c contributes its CS to the C^N set. The value function $\omega(s)$ maps from the set of neighbour CS C^N to the new CS C for a given species s. The value function is used to simulate an amount that is distributed amongst its neighbours in a time step Δt_i. In general the function $\omega(s)$ takes all the values c_s of a specific species with the concentration collection function $\sigma(s)$, subsequently a rule set has to be given in order to calculate the value of this species in the next time step. For example, the arithmetic mean of the neighbouring cells can be computed with:

$$\omega(s) = \frac{1}{|\sigma(C^N, s)|} \sum_{c_i(s) \in \sigma(C^N, s)} c_i(s) \tag{2.5}$$

The different value functions ω can also be called **value rules**.

2.2 Modelling of diffusion

2.2.1 Discretised diffusion

The diffusion of a species S can be simulated with a graph automaton with extended graph automaton cells. The initial concentration of the compound is given as a value in the current CS C at the time t_0 of an EGAC. Different value rules can then be applied to simulate an diffusion process through the GA.

An important aspect in modelling this phenomenon is the law of conservation of mass [de Lavoisier, 1801], which states that the mass in any closed system has to be constant over time. This means that the entire mass (the overall concentrations of each species) in the automaton needs to be equivalent at any time t_i, if no chemical changes occur. That leads to the following constraint:

$$c_{ges} = \sum_{v \in V} c_v = \text{const.} \; \forall \, t_i \; \forall \, S \tag{2.6}$$

First the continuous partial differential equation 1.3 has been translated to a recurrence relation in order to work with the discrete model of graph automata. In order to do this, the given differential equation has been approximated with differences. Therefore, the infinitesimal small time step ∂t and distance ∂x have been approximated with Δt and Δx and the concentration was rewritten to $c(t,x)$, to describe c as a function depending on

time and position in the graph:

$$\underbrace{\frac{1}{\Delta t}\big(c(t+\Delta t,x)-c(t,x)\big)}_{\text{change of } c \text{ per } \Delta t} = -D\frac{\partial}{\partial x}\underbrace{\frac{1}{\Delta x}\big(c(t,x+\Delta x)-c(t,x)\big)}_{\text{change of } c \text{ per } \Delta x} \tag{2.7}$$

By calculating the limit with $\Delta x, \Delta t \to 0$ of this function, it is possible to reobtain Fick's second law [Hütt, 2001]. The remaining differential quotient ∂x has also been rewritten with the neighbour relations of the graph in mind. Consequently, it was necessary to define what is near to the current position x, because the concentration is dependent on the immediate environment of a node v_n. Let $N = \{n_1, \cdots, n_m\}$ be the set of nodes that are adjacent to a node v_n with $m = |N|$. The position x can now be represented as a node v_n and $n \in N$ are nodes, that have a distance of Δx to v_n. Therefore, $c(t,x)$ needs to be rewritten again to $c(t,v)$, a function describing the concentration at a given node v at time t:

$$\begin{aligned}\frac{1}{\Delta t}\big(c(t+\Delta t,v_n)-c(t,v_n)\big) &= -D\frac{1}{\Delta v_n^2}\cdot \\ &\Big(\big(c(t,n_1)-c(t,v_n)\big)+\big(c(t,n_2)-c(t,v_n)\big)+\cdots+\big(c(t,n_m)-c(t,v_n)\big)\Big) \\ &= -D\frac{1}{\Delta v_n^2}\cdot\sum_{n_i\in N}\big(c(t,n_i)-c(t,v_n)\big)\end{aligned} \tag{2.8}$$

Since the space and time dimensions of the graph automata are uniform, it is possible to assume $\Delta t = 1$ and $\Delta v_n = 1$. By applying those rules, the previous equation results in:

$$c(t+1,v_n)-c(t,v_n) = -D\cdot\sum_{n_i\in N}\big(c(t,n_i)-c(t,v_n)\big) \tag{2.9}$$

and by rearranging further one obtains

$$c(t+1,v_n) = D\cdot\sum_{n_i\in N}c(t,n_i)+(1-m\cdot D)\cdot c(t,v_n) \tag{2.10}$$

Now it is possible to translate the equation to a value rule. The new CS C_t of an EGAC can be calculated with:

$$\omega(S) = D\cdot\sum_{c_i(S)\in\sigma(C^N,S)}c_i(S)+(1-|\sigma(C^N,S)|D)\cdot c_t(S). \tag{2.11}$$

The rule represents the discretised version of Fick's Law of diffusion in the context of graph automata.

2.2.2 Diffusion coefficient

A number of approaches to approximate the diffusion coefficient is given in Section 1.2.2. A different approach is to calculate the diffusion coefficient based on the molecu-

lar weight of a species. Molar mass and molar volume are tightly correlated (see Figure 2.3) because the molar volume is an assembled unit, calculated with $V = {}^{M}/_{\rho}$, where ρ is the volumetric mass density of the substance. On the other hand the modified STOKES-EINSTEIN equation [Von Smoluchowski, 1906] suggests that the diffusivity of a solute depends directly on the Stokes radius R_0 of a species,

$$D = \frac{k_\mathrm{B} \cdot T}{6 \cdot \pi \cdot \eta \cdot R_0} \tag{2.12}$$

which is represented with the molar volume in the previous correlations. The regression of molar volume against molecular mass shows no significant outliers and a coefficient of determination of $R^2 = 0.938$, which indicates a good correlation.

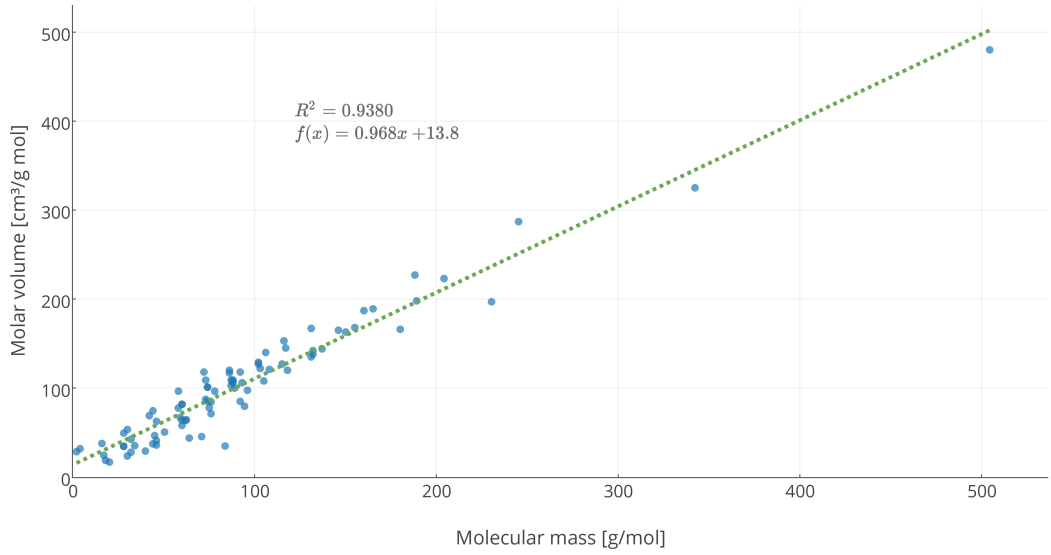


Figure 2.3: Depicted are the molar volumes against the diffusion coefficient of 86 species. The molar masses of the substances have been fetched from the Chemical Entities of Biological Interest (ChEBI) database [Degtyarenko et al., 2008]. The molar volumes have been taken from [Hayduk and Laudie, 1974]. A linear regression has been calculated and is shown as a dashed green line.

Another important factor is that the molecular weight can be easily calculated from the molecular structure of a species. But the molar volume is subject to variation because of different influencing factors, such as polarity and hydrophobicity of both solvent and solute. Therefore, most databases only hold values for the molar mass of a component but not its molar volume. For the implementation of the application it is important that the values that are used in the calculations of the simulation are easily accessible and correct. Therefore, an additional correlation connecting molar mass with diffusivity has been calculated with R [R Development Core Team, 2008] and can be seen in Figure 2.4. The correlation does not produce a fit as good as the correlations that have been proposed previously using molar volume, with a coefficient of determination of $R^2 = 0.8715$. Evidently it is possible to estimate the diffusivity with the molar volume, but because the values are more scattered, the correlation cannot be as good. To apply

multidimensional curve fitting including viscosity and temperature, the availability of data that has been experimentally validated is too sparse.

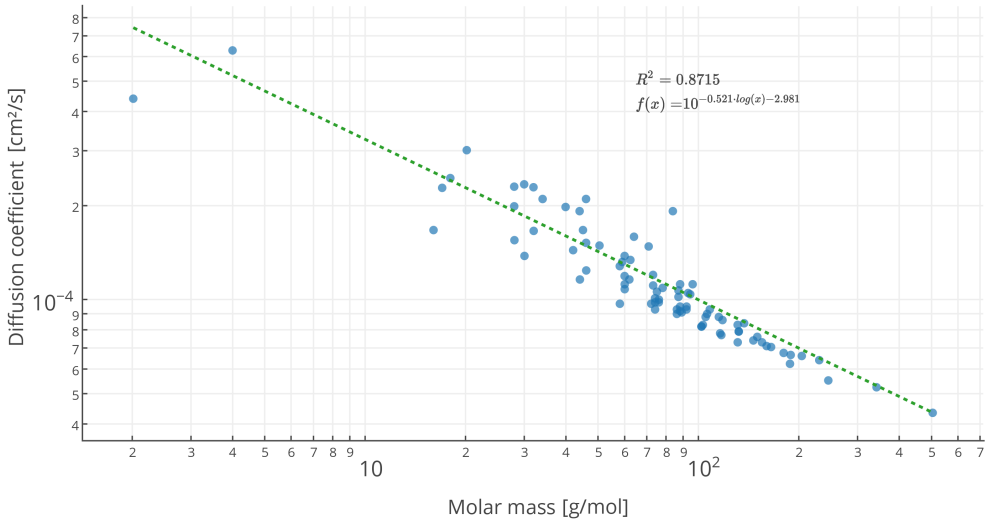


Figure 2.4: Depicted are the molar masses against the diffusion coefficients of 86 species. The diffusion coefficients have been taken from [Hayduk and Laudie, 1974]. The molar masses of the substances have been fetched from the Chemical Entities of Biological Interest (ChEBI) database [Degtyarenko et al., 2008]. Additionally, the proposed correlation is displayed as dashed line in the plot.

It is apparent that no single correlation can exactly model the diffusion constant. A lot of factors influence the speed of diffusion and the accurate inclusion of all these factors, that also influence each other, is hard to accomplish. Moreover, if more factors were considered the applicability of the model is lowered, because it has to be presumed that all the data is available for all the species and environments that could be used. Furthermore, the available data from the years where the estimation of the diffusion coefficient was prominent is not consistent. Multiple solutions, temperatures, and viscosities have been measured but not with standardized parameters and methods. Additionally it is difficult to determine the diffusivity of a species without including its actual three-dimensional extent to accurately describe the flow dynamics.

It is, however, possible to give an estimation of the diffusivity from the molar volume or molar mass, that is able to represent the actual value of a species accurately. In this implementation of the proposed model the diffusion is calculated using two different methods. The diffusion constant of proteins and species with a mass of more than $10,000\ {}^{g}/_{mol}$ is calculated with the YOUNG correlation. In case the species is lighter, the diffusion coefficient is approximated with the WILKE correlation, where the molar volume has been approximated by molar mass. In cellular environments the resulting coefficient will be multiplied with the STDF of ${}^{D_{cyto}}/_{D_{water}} = 0.27$ as described by [Kao et al., 1993].

2.2.3 Dimensional scaling

While the distance- and time scale of GA are dimensionless and uniform, in reality it is not. The diffusion coefficient D is the only constant in the value rule $\omega(s)$ (see equation 2.11) that connects the model with specific characteristics of nature, and therefore needs to be parametrized. As described, the diffusion coefficient D will be estimated depending on the molecular weight of the respective species. The coefficient that is calculated with the correlations is always in cm^2/s. To apply the respective calculation to the GA, the distance between two nodes with distance Δd and the time Δt between two time steps needs to be specified before the beginning of the simulation. After the estimation of D the constant needs to be converted to the required unit. The result will be denoted by D^c. Subsequently, the quantity of Δd and Δt has to be considered. Therefore the adjusted diffusion coefficient D^* can be obtained as following:

$$D^* = D^c \cdot \frac{\Delta d^2}{\Delta t} \tag{2.13}$$

A constraint is, that the resulting diffusion constant has to be small compared to the spacial and temporal discretisation. Simply put, this means that the adjusted diffusion constant D^* needs to be smaller than the current concentration of a species. In case D^* is close to 1 the propagation of the diffusion is becoming less accurate. D^* must not be larger than 1. Accordingly, the dimensions should be chosen so high that the diffusion coefficient is small enough. This can be achieved by choosing a larger Δd or a smaller Δt.

2.3 Modelling of chemical kinetics

Again, the partial differential equations have to be translated to recurrence relations, in order to apply to the discrete model of graph automata. The transformation starts by translating the reaction rate v to the change of concentration $dc(S)$ over a time dt. Subsequently, those differentials will be approximated with $\Delta c(S)$ and Δt:

$$v = \frac{\Delta c(S)}{\Delta t} = \frac{1}{h(P)}\frac{\Delta c(P)}{\Delta t} = -\frac{1}{h(A)}\frac{\Delta c(A)}{\Delta t} = kc(A) \tag{2.14}$$

Further, the concentration will be rewritten to $c(S,t)$, to describe c as a function depending on the species and time:

$$\frac{1}{\Delta t}\Delta c(S,t+\Delta t) = \frac{1}{h(P)}\frac{1}{\Delta t}\Delta c(P,t+\Delta t) = -\frac{1}{h(A)}\frac{1}{\Delta t}\Delta c(A,t+\Delta t) = kc(A,t) \tag{2.15}$$

Now, because the uniformity of the time steps, $\Delta t = 1$ can be assumed which leads to:

$$\Delta c(S,t+1) = \frac{1}{h(P)}\Delta c(P,t+1) = -\frac{1}{h(A)}\Delta c(A,t+1) = kc(A,t) \tag{2.16}$$

Therefore the recurrence relation for the new concentration of a species S at $t+1$ in a first-order reaction can be calculated with:

$$c(S,t+1) = \begin{cases} c(S,t) + k \cdot h(S) \cdot c(A,t) & \text{if S is a product,} \\ c(S,t) - k \cdot h(S) \cdot c(A,t) & \text{if S is a substrate.} \end{cases} \tag{2.17}$$

Using the same method for second order reactions, the resulting recurrence relations are:

$$c(S,t+1) = \begin{cases} c(S,t) + k \cdot h(S) \cdot c(A,t)^{\alpha} \cdot c(B,t)^{\beta} & \text{if S is a product,} \\ c(S,t) - k \cdot h(S) \cdot c(A,t)^{\alpha} \cdot c(B,t)^{\beta} & \text{if S is a substrate.} \end{cases} \tag{2.18}$$

The following equations represent the recurrence relations of reversible reactions for the application with graph automata:

$$c(S,t+1) = \begin{cases} c(S,t) - h(S) \cdot (k_1 \cdot c(A,t) - k_{-1} \cdot c(B,t)) & \text{if S is a substrate} \\ c(S,t) + h(S) \cdot (k_1 \cdot c(A,t) - k_{-1} \cdot c(B,t)) & \text{if S is a product.} \end{cases} \tag{2.19}$$

In this work the concentrations of both enzyme and substrate are given and can vary due to diffusion or possible use in other metabolic pathways. Therefore, the equation where the amount of enzyme is not negligible (Equation 1.22) has been used as a starting point to derive the following recurrence relation:

$$c(S,t+1) = \begin{cases} c(S,t) + h(S) \cdot \frac{k_{cat} \cdot c(E,t) \cdot c(A,t)}{k_m + c(A,t)} & \text{if S is a product,} \\ c(S,t) - h(S) \cdot \frac{k_{cat} \cdot c(E,t) \cdot c(A,t)}{k_m + c(A,t)} & \text{if S is a substrate.} \end{cases} \tag{2.20}$$

In chemical kinetics the constant, that joins model with reality is the rate constant k. First order rate constants are expressed as a kind of frequency (e.g. $1/s$ – once per second), whereas second order rate constants are represented as a concentration multiplied by a frequency (e.g. $mol/l \cdot s$ – mole per litre and second) [Cornish-Bowden, 2013]. Hence a similar principle to the coupling of scales in diffusion can be used. The space between two nodes is not required, only the time step is needed to scale the constant. First the unit of the constant is converted to the specified unit given by the system and denoted by k^c. The adjusted k^* is than calculated by $k^* = k^c \cdot \frac{1}{\Delta t}$ in both cases.

3 Implementation

The model has been implemented in order to verify its applicability and to promote its usage. The first and fundamental layer of the system is the modelled structure of the GA (Section 2.1). The mathematical principles have been encapsulated in extensible program-code-templates (called classes) that are also extensible, in case the model needs to be broadened or modified in the future. On top of the first layer a second layer represents the concrete implementation of the suggested phenomena (namely diffusion and chemical kinetics). The third layer encapsulates the input of parameters, species, and scaling that govern the processes at the second layer. To enable easy integration of data, interfaces to import data from multiple databases are provided. The data that is needed to scale the system is given by the user. All of the layers are accessible via the application programming interface (API). Additionally, a graphical user interface (GUI) has been implemented to provide a minimalistic set of features to create, simulate, and observe graph automata.

The **G**raph **A**utomata **Si**mulation (GASi) Java API has been developed and implemented in the Java programming language using the 1.8 Java Runtime Environment. The **Jersey** RESTful Web Services framework (Version 2.17) is an open source framework for developing RESTful Web Services. GASi uses the Jersey implementation of the Java API for RESTful Web Services (JAX-RS) to interact with the SABIO-RK Database. The **ChEBI API** by the European Bioinformatics Institute provides programmatic access to a database of Chemical Entities of Biological Interest that is here used to retrieve chemical entities (called species in this work). The **Unit of Measurement** (UoM) library provides a set of programming interfaces for handling units and quantities. The version 0.8 API and reference implementation (RI) of UoM are important essentials to the dimensional scaling and unit management in GASi. The **JavaFX** (Version 8) technology has been used as a basic set of graphics and media packages that advance the development of GUI.

The provided API includes the following features:

- the creation of graphs
- the creation of automata
- the definition of environmental variables (providing dimensional scaling)
- an implementation of the reaction model (first-order reactions, second-order reactions, equilibrium reactions, enzymatic reactions)
- an implementation of the diffusion model
- the simulation of automata (with diffusion and / or chemical reactions)

- an implementation to render graphs on a JavaFX Canvas
- an interface to apply an implementation (provided by ZHENYU PAN) of the sweepline algorithm for VORONOI diagrams [Fortune, 1987]
- an implementation of the FRUCHTERMANN and REINGOLD graph drawing algorithm [Fruchterman and Reingold, 1991]
- an implementation to save and load graphs to and from GraphML [Brandes et al., 2002] files
- an interface to integrate data from the ChEBI database using the provided Java implementation [Degtyarenko et al., 2008]
- an implementation to integrate data from the PubChem database [Bolton et al., 2008] database
- an implementation to integrate data from the SABIO-RK Database [Wittig et al., 2012] database.

3.1 Data model

The primary classes and their relations in the GASi API are illustrated in Figure 3.1. Class names are typeset in typewriter font to distinguish them from regular terminology. The `GraphAutomata` relies on four main components, namely the `Graph`, a list of `Reactions`, the `RecurrenceDiffusion`, and the `EnvironmentalVariables`. The following listing of components tries to give a brief overview. For the full information, please refer to the source code and documentation.

3.1.1 Graph

The `Graph` can be used with generic `Nodes` and `Edges` or with specialized `BioNodes` and `BioEdges`, that can be given as parameterized types. In this implementation of a graph, a model, based on the adjacency list, has been chosen. It combines a relatively low space requirement with fast linear searches. This compromise is acceptable in the context of this model. An overview of the possible graph representations can be seen in Table 1.2.5. The complexity of a search in an adjacency list is dependent on the average degree $deg(v_i)$ of a node v_i. In the model the degree of a node represents the number of connections to the neighbours of the cell. The average degree of a node will be significantly smaller than the number of nodes or edges. Therefore, this type of representation is superior to the edge list representation. The adjacency matrix is practically not usable because of exponentially growing space requirements. Consequently, each `Node` holds a list of references to its neighbours, in addition to a `Vector2D`, that defines its position in a `Space2D`. `BioNodes` give the possibility to assign a set of `Species` with concentrations to each node. `BioEdges` offer the possibility to add a permeability for species to an edge. The permeability is currently not used in this implementation but it could be utilized for example to hinder diffusion at membranes or other obstacles.

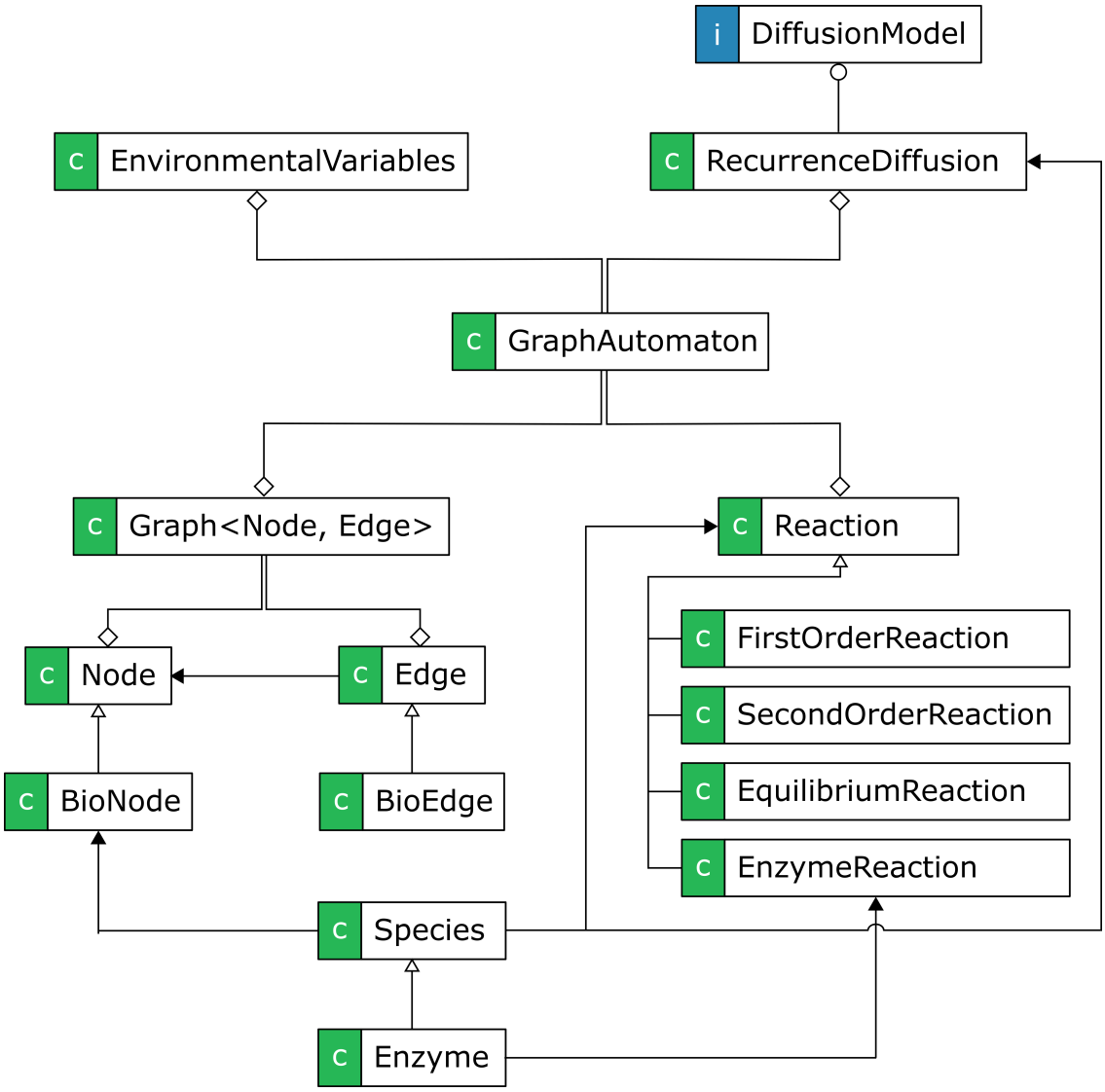


Figure 3.1: The schematic representation shows the data model of GASi. Boxes symbolise classes (in green) and interfaces (in blue). Lines show the connections between classes. Empty triangles (▷) represent inheritance, full triangles (▶) represent association, diamonds (◇) represent composition, and circles (○) illustrate an implementation of an interface.

3.1.2 Kinetics

The `Reaction` class is an abstract class that provides the basis for the chemical kinetics implementation. Each reaction has two sets of `Species`, that define its substrates and products, a mapping from those species to their stoichiometric coefficients, and a mapping of the reaction orders. Each reaction subclass has to define a method to update the concentrations in a `BioNode`. `FirstOrderReactions` additionaly define a `FirstOrderReactionRate` that is used to calculate the next concentration of the components based on the species and stoichiometric coefficients. `SecondOrderReactions` equally define a `SecondOrderReactionRate` and a mapping for the reaction orders that

are needed for the calculation of the updated concentrations. `EquilibriumReaction` define two `SecondOrderReactionRate`s for forward and backward rate constants. Enzyme kinetics based on Michaelis-Menten kinetics can be simulated with the class `EnzymeReaction`, that uses an `Enzyme` with defined k_{cat} and v_{max} to simulate the reactions. A new `Reaction` can be added to the automata and will be applied to every `BioNode` in the graph, every time a time step is performed.

3.1.3 Diffusion

The interface `DiffusionModel` can be used to implement a method for the calculation of concentrations of species in the given node that are attributable to diffusion. In contrast to the updates from `Reaction`s, which are instantaneous, the updates of diffusion are collected first and only when every species in every node is calculated the update is propagated to the graph. The `RecurrenceDiffusion` holds the diffusion coefficients of all the species. Those coefficients are calculated once, with implementations of the suggested correlations, after all species have been added to the graph and before the simulation begins.

3.1.4 Environmental variables

The class `EnvironmentalVariables` is used to store all variables that are used all over the system. Every variable is represented by a the specified unit and its quantity. Those variables consist of the node distance as a quantity of length, the time step as a quantity of time, the temperature, the dynamic viscosity of the system, and an boolean value if the whole system was considered as a cellular enviornment. All the calculations in kinetics and diffusion are scaled with those variables.

3.2 Data integration

Valuable data can be retrieved from a variety of sources. In this implementation a general parser pattern has been created to integrate data from a local or online source. Given a resource, which could be a local path or entity identifier, each parser can be implemented to handle this resource differently but construct the same object as output.

Each class with the suffix `ParserService` symbolizes an implementation of this pattern. In the current implementation this classes are:

- `ChEBIParserService`
  Gets a `String` which contains the ChEBI Identifier of the compound of interest. A call of the `fetchSpecies()` method fetches the chemical entity online and returns the `Species`.

- `GraphMLParserService`
  Gets a `String` which contains the path to an XML file in GraphML format. A call of the `fetchGraph()` method parses the XML file and returns the `Graph` with `BioNodes` and `BioEdges`.
- `PubChemParserService`
  Gets a `String` which contains the path to an XML file in the PubChem format. A call of the `fetchSpecies()` method parses the XML file and returns the `Species`.
- `SabioRKParserService`
  Gets a `String` which contains the Kinetic Law Identifier as specified by the SABIO-RK database. A call of the `fetchReaction()` method fetches the kinetic law entity online and returns the `EnzymeReaction`, if possible. In order to convert the law to a `EnzymeReaction` the entry must specify k_{cat} and k_m values. The required species are automatically retrieved from ChEBI.

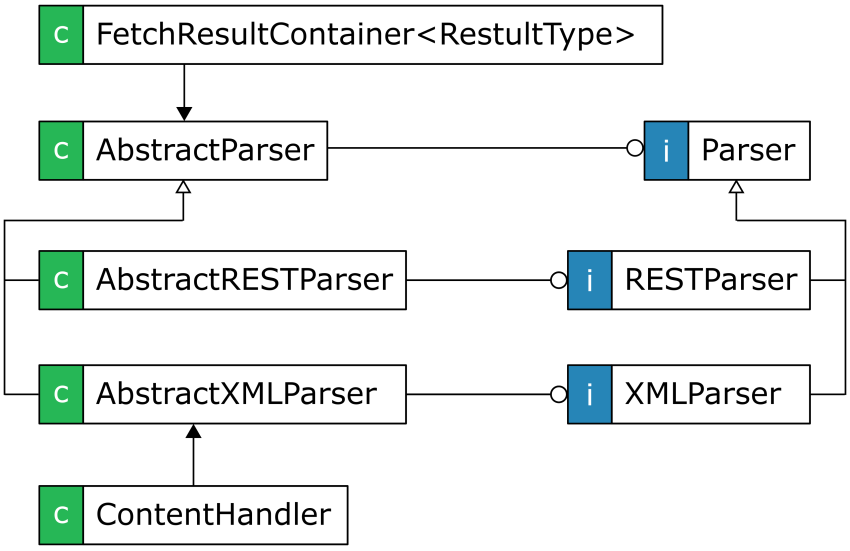


Figure 3.2: The schematic representation shows the parser model for data integration of GASi. Boxes symbolize classes (in green) and interfaces (in blue). Lines show the connections between classes. Empty triangles (▷) represent inheritance, full triangles (▶) represent association, and circles (○) illustrate an implementation of an interface.

3.3 Algorithms

The algorithm for the diffusion (Algorithm 1) has to return a mapping $f : S \mapsto c$ where each species S is mapped to a concentration c. These mappings are collected for each node and are propagated to the graph, only when every species in every node is calculated. The diffusion coefficient is calculated in a separate Algorithm 2. To improve the performance of the diffusion algorithm all diffusion coefficients are calculated once at the start of the simulation.

Algorithm 1: Recurrence diffusion update

Input: BioNode *N*

Output: A mapping from a Species *S* to a concentration *c* $f : S \mapsto c$

foreach *Species S* **in** *N.Species* **do**
  $d \leftarrow$ `calculateDiffusionCoefficient(S)`
  $nc \leftarrow 0.0$
  foreach *BioNode NN* **in** *N.Neighbours* **do**
    $nc \leftarrow nc + NN.S.\text{concentration}$
  $tc \leftarrow N.S.\text{concentration}$
  $nextc \leftarrow d \cdot nc + (1 - |N.\text{Neighbours}| \cdot d) \cdot tc$
  add $S \mapsto nextc$ to the mapping f
return f

Algorithm 2: Calculate diffusion coefficient

Input: Species *S*, EnvironmentalVariables *V*

Output: The diffusion coefficient *d*

Temperature *temp* $\leftarrow$ V.Temperature
Viscosity *visc* $\leftarrow$ V.Viscosity
Length *nd* $\leftarrow$ V.NodeDistance
Time *ts* $\leftarrow$ V.TimeStep
if $S.Mass < 10,000$ **then**
  d $\leftarrow \frac{7.48 \cdot 10^{-8} \cdot (2.26 \cdot 18.0153)^{0.5} \cdot temp.value}{\text{visc.value} \cdot (0.968 \cdot S.\text{Mass} + 13.8)^{0.6}}$
else
  d $\leftarrow \frac{8.34 \cdot 10^{-8} \cdot temp.value}{\text{visc.value} \cdot S.\text{Mass}^{1/3}}$
if *V.isCellularEnvironment* **then**
  d $\leftarrow$ d $\cdot 0.27$
Diffusivity dif $\leftarrow$ d **as** cm^2/s
dScaled $\leftarrow$ *d* **to** $nd.unit^2/ts.unit$
dScaled $\leftarrow$ *d* $\cdot \frac{nd.value^2}{ts.value}$
return *dScaled*

The progression of the different reaction types are calculated very similar. First the reactions rate constants are scaled with the defined time step. Afterwards, the reaction rates are calculated using the formulas that have been deduced in Chapter 2.3. Subsequently, the concentrations of the species are adjusted accordingly.

The velocity of first-order reactions (see Algorithm 3) are only dependent on a single rate determining substrate. Still, there can be more species involved in the reaction. Therefore, their concentrations are adjusted accordingly with the given velocity.

Second-order reactions (see Algorithm 4) are dependant on two substrates and the reaction rate that is often determined experimentally. Similar to first-order reaction the concentrations of all substrates and products given are updated, even if they are not used to calculate the velocity.

Reversible reactions are (see Algorithm 5) are faster when the substrates outweigh the products and slower vice versa. The reaction strives towards an equilibrium, that is specified by the quotient of the forward and backward reaction rate.

The progression of the Michaleis-Menten reaction (see Algorithm 6) is specified by the k_m and k_{cat} values, that are enzyme and substrate specific. Lower k_m and higher k_{cat} increase the overall velocity of the reaction.

Algorithm 3: First-order reaction update

Input: BioNode *N*, EnvironmentalVariables *V*
Output: void

1 Time *ts* ← V.TimeStep
2 Species *substrate* ← Reaction.DeterminingSubstrate
3 RateConstant *k* ← Reaction.RateConstant
4 *kScaled* ← *k* **to** $1/ts.unit$
5 *kScaled* ← *k* $\cdot 1/ts.value$
6 *v* ← *kScaled* · *N.substrate*.concentration
7 **foreach** *Species S* **in** *Reaction.Substrates* **do**
8 ⌊ *N.S*.concentration ← *N.S*.concentration − *v* · Reaction.StoichiometricCoefficient.S
9 **foreach** *Species P* **in** *Reaction.Products* **do**
10 ⌊ *N.P*.concentration ← *N.P*.concentration + *v* · Reaction.StoichiometricCoefficient.P

Algorithm 4: Second-order reaction update

Input: BioNode *N*, EnvironmentalVariables *V*
Output: void

1 Time *ts* ← V.TimeStep
2 Species *substrateA* ← Reaction.DeterminingSubstrateA
3 Species *substrateB* ← Reaction.DeterminingSubstrateB
4 RateConstant *k* ← Reaction.RateConstant
5 *kScaled* ← *k* **to** $1/mol \cdot ts.unit$
6 *kScaled* ← *k* $\cdot 1/mol \cdot ts.value$
7 *v* ← *kScaled* · *N.substrateA*.concentration$^{Reaction.Rates.substrateA}$ · *N.substrateB*.concentration$^{Reaction.Rates.substrateB}$
8 **foreach** *Species S* **in** *Reaction.Substrates* **do**
9 ⌊ *N.S*.concentration ← *N.S*.concentration − *v* · Reaction.StoichiometricCoefficient.S
10 **foreach** *Species P* **in** *Reaction.Products* **do**
11 ⌊ *N.P*.concentration ← *N.P*.concentration + *v* · Reaction.StoichiometricCoefficient.P

Algorithm 5: Reversible reaction update

Input: BioNode *N*, EnvironmentalVariables *V*
Output: void

Time *ts* ← V.TimeStep
Species *substrate* ← Reaction.Substrate
Species *product* ← Reaction.Product
RateConstant *kf* ← Reaction.ForwardRateConstant
kfScaled ← *kf* **to** $1/ts.unit$
kfScaled ← *kf* $\cdot 1/ts.value$
RateConstant *kb* ← Reaction.BackwardRateConstant
kbScaled ← *kb* **to** $1/ts.unit$
kbScaled ← *kb* $\cdot 1/ts.value$
$v \leftarrow kfScaled \cdot \prod_{\text{S in Reaction.Substrates}}(N.S.\text{concentration}) - kbScaled \cdot \prod_{\text{P in Reaction.Products}}(N.P.\text{concentration})$
foreach *Species S* **in** *Reaction.Substrates* **do**
    $N.S.\text{concentration} \leftarrow N.S.\text{concentration} - v \cdot \text{Reaction.StoichiometricCoefficient.S}$
foreach *Species P* **in** *Reaction.Products* **do**
    $N.P.\text{concentration} \leftarrow N.P.\text{concentration} + v \cdot \text{Reaction.StoichiometricCoefficient.P}$

Algorithm 6: Michaleis-Menten reaction update

Input: BioNode *N*, EnvironmentalVariables *V*
Output: void

Time *ts* ← V.TimeStep
Species *substrate* ← Reaction.DeterminingSubstrate
Species *enzyme* ← Reaction.Enzyme
RateConstant *kCat* ← enzyme.KCat
RateConstant *km* ← enzyme.KM
kCatScaled ← *kCat* **to** $1/ts.unit$
kCatScaled ← *kCat* $\cdot 1/ts.value$
$v \leftarrow \frac{kCatScaled \cdot N.enzyme.\text{concentration} \cdot N.substrate.\text{concentration}}{km + N.substrate.\text{concentration}}$
foreach *Species S* **in** *Reaction.Substrates* **do**
    $N.S.\text{concentration} \leftarrow N.S.\text{concentration} - v \cdot \text{Reaction.StoichiometricCoefficient.S}$
foreach *Species P* **in** *Reaction.Products* **do**
    $N.P.\text{concentration} \leftarrow N.P.\text{concentration} + v \cdot \text{Reaction.StoichiometricCoefficient.P}$

4 Results and Discussion

4.1 Diffusion

To asses if the simulated diffusion as calculated by the implementation of the suggested model was viable, it is compared to the classical approach in order to solve this problem. This can be done by calculating the average displacement of a particle under the influence of Brownian motion [Metzler and Klafter, 2004]. Cellular diffusion is more difficult to describe, because many factors influence the speed of a particle in a cellular environment (see Section 2.2.2). As a sample system to evaluate the cellular diffusion a fluorescence recovery after photobleaching (FRAP) experiment has been re-staged and the results of simulation and experiment have been compared.

4.1.1 Classical diffusion

To evaluate the diffusion, a sample GA system has been created. The rectangular graph contains 5×10 nodes. The inner nodes have four neighbours each, the corners have two, and the edges of the graph have three neighbours. This structure is equivalent to a two-dimensional cellular automaton with squared tiling and the VON NEUMANN neighbourhood with radius 1. The graph automaton is depicted in Figure 4.1. The four species Methanol, Ethylene glycol, Valine, and Sucrose have been added to the GA. The five rows of nodes on the left side have been set to a concentration of 1 $^{mol}/_{l}$ for each of the species, the five on the right have a concentration of 0 $^{mol}/_{l}$ respectively. The environmental variables have been set to: a time step $\Delta t = 1\ \mu s$, the node distance $\Delta d = 250\ nm$, the dynamic viscosity $\eta = 1\ mPas$, and the temperature $T = 293K$. Water has been chosen as the solvent molecule. The observed node (see Figure 4.1) is 750 nm away from the next node, that contains a high concentration.

In theory, the displacement of a particle under the influence of Brownian motion can be calculated with the formula [Metzler and Klafter, 2004]:

$$\langle x^2(t) \rangle = 2n \cdot D \cdot t \tag{4.1}$$

where

- $\langle x^2(t) \rangle$ is a measure for the distance a particle travels on average in the time t
- n is the number of spacial dimensions, and
- D is the diffusion coefficient.

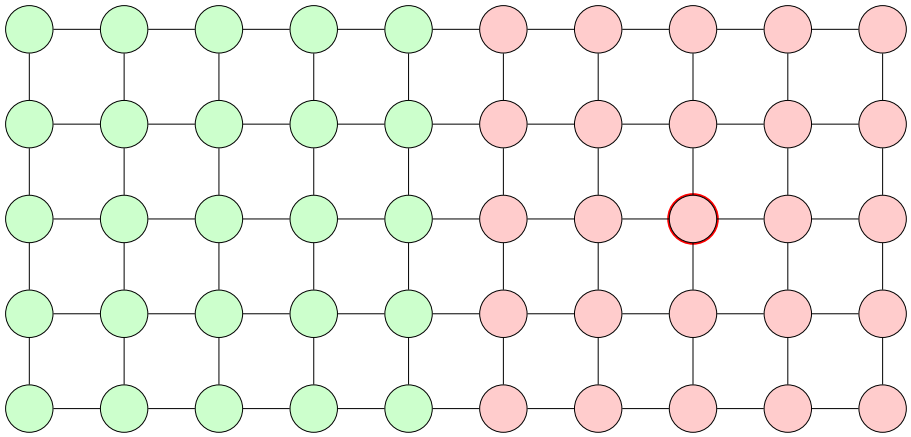


Figure 4.1: The diffusion graph automaton is depicted. The nodes are colour coded with the concentration of the species - green is depicting a concentration of 1 $^{mol}/_{l}$ and red a concentration of 0 $^{mol}/_{l}$. The node with the red border will be observed while simulating the graph automaton.

After rearranging the equation, the average time a particle takes to move a certain distance x can be approximated by:

$$t = \frac{x^2}{2n \cdot D} \tag{4.2}$$

In case of graph automata the dimensionality of the equation is one ($n = 1$), because the diffusion is directed via edges and therefore has only one degree of freedom.

Further, the GA has been simulated to observe the half-life time, which the different species need to reach the observed node. In this context, the half-life time $\tau_{1/2}$ is the time required for half of the concentration of the steady state to arrive [Kang et al., 2012] at the specified node. Figure 4.2 shows the concentrations of the species over the course of the simulation.

The total simulation duration has been $50,000$ time steps ($0.05\ s$). The system has been simulated twice, once with the approximated values for the diffusion coefficients as calculated by the correlation D_{Ap} and once with the literature values for the diffusion coefficients D_{Ex} [Hayduk and Laudie, 1974] (see Table A.1) for the respective species. Afterwards the half-life times T_{Si}^{Ap} for the simulation with approximated D_{Ap} and T_{Si}^{Ex} for the simulation with experimental D_{Ex} have been compared with the calculated T_{Ca}^{Ex} (using equation 2.11) using experimental D_{Ex}. The resulting data is shown in the Table 4.1.

The time needed for the species to diffuse is approximated in the right order of magnitude (results in Table 4.1). Small molecules and experimental values show a better overall correlation than heavy molecules and approximated coefficients. Still, there is a lot of room for optimization. It seems that in general the diffusion is slower in the simula-

tion and with a doubled weight, while the divergence in time is also doubled. This could be bypassed by parametrization of the diffusion equation or the diffusion coefficient, but ideally all equations and modelled processes should be self-contained and correct if the foundations were reliable. Further verification and optimisation of the model could be capable to reduce the error to a certain extent.

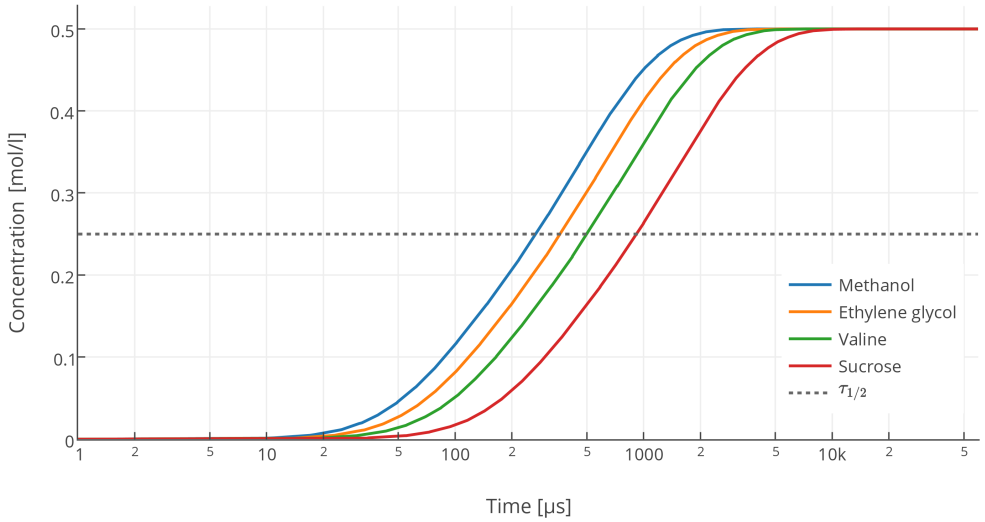


Figure 4.2: Depicted are the concentrations of small molecules Methanol, Ethylene glycol, Valine and Sucrose in water over the course of the simulation. The half-life time $\tau_{1/2}$ is indicated by the dashed horizontal line. The underlying diffusion coefficients have been calculated with Wilke correlation [Wilke and Chang, 1955].

Table 4.1: The simulated diffusion time in comparison to the calculated diffusion time. The travel distance is $750\ nm$.

Species	**Methanol**	**Ethylene glycol**	**Valine**	**Sucrose**
D_{Ap} in cm^2/s	1.41E-05	1.04E-05	7.55E-06	4.15E-06
D_{Ex} in cm^2/s	1.66E-05	1.16E-05	7.70E-06	5.24E-06
T_{Ca}^{Ex} in μs	169	242	365	536
T_{Si}^{Ap} in μs	271	365	504	917
Error in %	60.36	72.78	82.25	225.44
T_{Si}^{Ex} in μs	230	328	495	728
Error in %	36.09	50.89	76.92	113.61

4.1.2 Cellular diffusion

The diffusion coefficient in cells can be measured with FRAP. The method became popular in the 1970th, when non-invasive fluorescent tagging became available using GFP. In principle the cell is injected with a fluorescent protein or is transformed with the corresponding gene. The adult cell is than targeted with a laser beam of a defined diameter. When the laser is focused on a certain area and shortly activated, the GFP in this region receives a high intensity illumination which causes their fluorescence lifetime to elapse quickly. Subsequently, the cell exhibits a dark spot, where the laser was used to bleach the cell. Now the centre of the bleach spot is constantly observed, and its fluorescence measured. Because the GFP in this region is permanently bleached the new emergent fluorescence can only be attributed to GFP that arrives there due to diffusion. For a illustration of the recovery curve see Figure 4.3 [Reits and Neefjes, 2001].

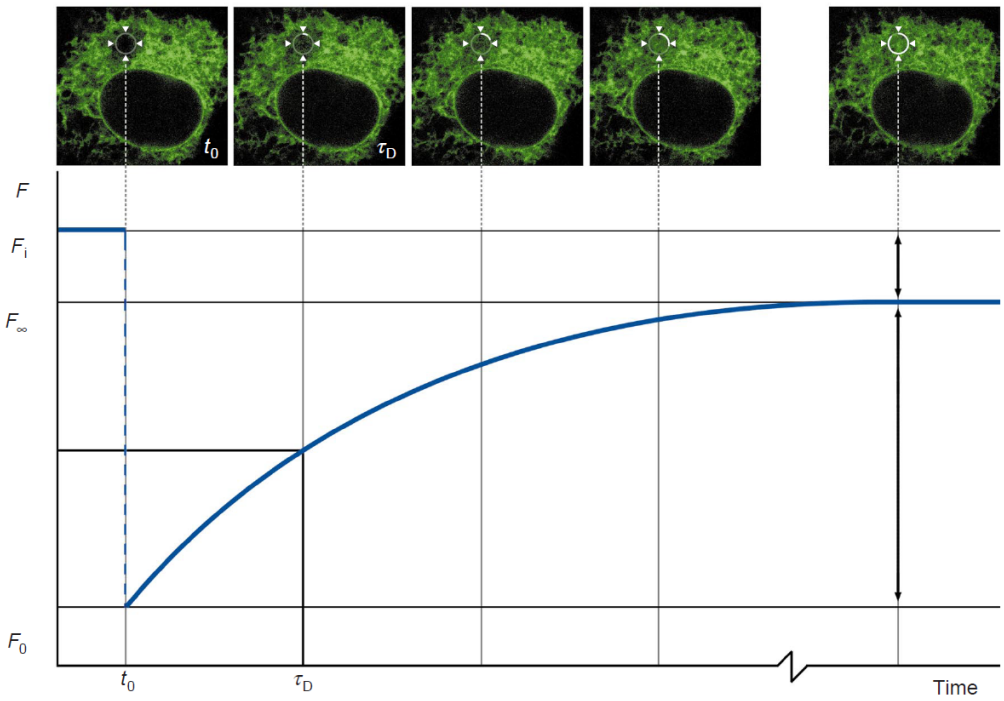


Figure 4.3: Fluorescence recovery after photobleaching. After bleaching a region at time t_0 the initial fluorescence F_i is decreased to F_0. The fluorescence recovers slowly up to an upper threshold F_∞. The half-life time $\tau_{1/2}$ marks the point, where half of F_∞ has recovered. Modified after [Reits and Neefjes, 2001].

As a case of application of FRAP the experiment conducted by SWAMINATHAN has been chosen [Swaminathan et al., 1997] . *Escherichia coli* BL21 were transformed with a plasmid coding for GFP-S65T (UniprotID: P42212). The photobleaching spot with a diameter of $5\ \mu m$ resulted in a half-life recovery time of 83 ± 6 ms. As a validation of the diffusion in cellular environments a graph automaton has been constructed to recreate the results measured by SWAMINATHAN.

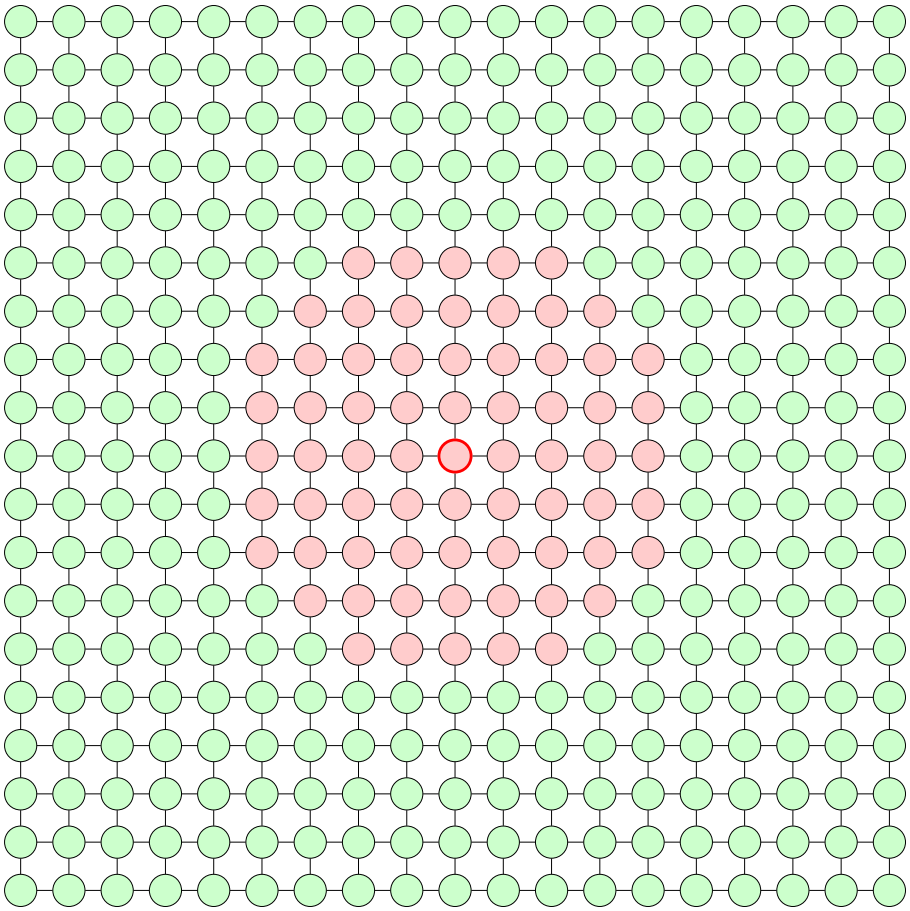


Figure 4.4: The cellular diffusion graph automaton is depicted. The nodes are colour coded with the concentration of the species – green depicting a concentration of 1 $^{mol}/_{l}$ and red a concentration of 0 $^{mol}/_{l}$. The node with the red border will be observed while simulating the graph automaton.

The rectangular graph contains 19×19 nodes (see Figure 4.4). Again inner nodes have four neighbours each, the corners have two and the edges of the graph have three neighbours. This structure is equivalent to a two-dimensional cellular automaton with squared tiling and the VON NEUMANN neighbourhood with radius 1. As a species GFP-S65T (26,886 $^{g}/_{mol}$) has been added to the GA. A circular sub-graph in the middle of the graph with a diameter of nine nodes has a concentration of 0 $^{mol}/_{l}$ and every other node has a concentration of 1 $^{mol}/_{l}$. The time step has been set to $\Delta t = 5\ \mu s$, the node distance to $\Delta d = 500\ nm$, the dynamic viscosity to $\eta = 1\ mPas$, and the temperature to $T = 293K$. The diameter of the circle is $5\ \mu m$, because the nodes with a concentration of 1 $^{mol}/_{l}$ are separated by 10 edges (with $\Delta d = 500\ nm$). Water has been chosen as the solvent molecule. The observed node is in the middle of the circular sub-graph.

The total simulation duration has been $100,000$ time steps ($0.1\ s$). The system has been simulated twice, once with the approximated values for the diffusion coefficients as calculated by the YOUNG correlation D_{Ap} and once with the literature value for the diffusion coefficients $D_{Ex} = 8,7 \cdot 10^{-7}$ for GFP in water, as calculated by [Swaminathan et al.,

1997]. The half-life times are $T_{Si}^{Ap} = 67,040\ ms$ for the simulation with approximated D_{Ap} and $T_{Si}^{Ex} = 65,455\ ms$ for the simulation with experimental D_{Ex}. The error for the simulation with approximated values is 19.22% – and slightly higher with 21.13% for the experimental value.

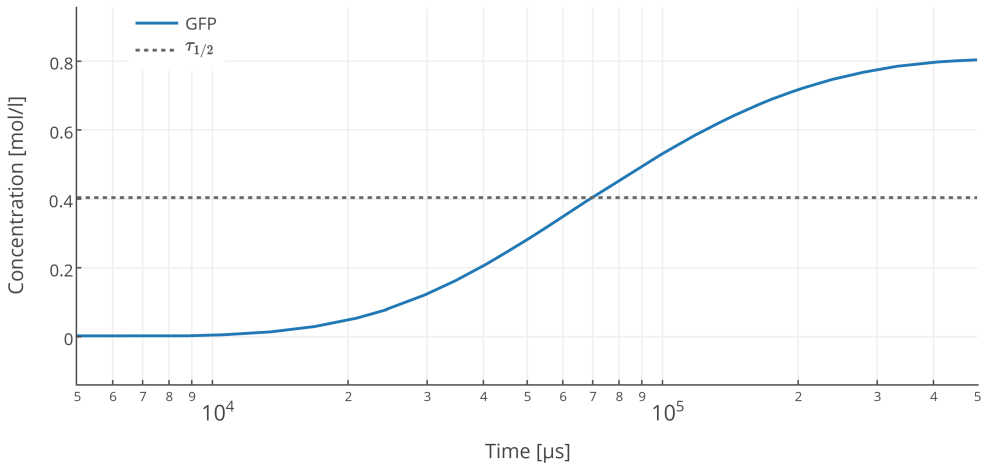


Figure 4.5: Depicted is the concentration of GFP in a cellular environment over the course of the simulation. The half-life time $\tau_{1/2}$ is indicated by the dashed horizontal line. The underlying diffusion coefficients have been calculated with the YOUNG correlation [Young et al., 1980].

The time needed for the GFP to diffuse is approximated in the right order of magnitude. An error of around 20% is on a percentage basis smaller than the ones obtained by classical diffusion. The underling YOUNG correlation seems to give a better estimation of the diffusion coefficient of proteins than the WILKE correlation for small molecules. Overall, this result could be the consequence of two primary factors. The YOUNG correlation was created for the estimation of diffusion coefficients based on molar masses, while the molar volumes needed for the WILKE correlation have to be approximated at first. Additionally, the system has been larger (higher number of nodes), it is possible that the size of the system is correlated with the accuracy of the prediction. Larger systems can give a better estimate because more sub-states or sub-systems can be used to predict the final phenomenon (see Introduction: 1.2.1 Holism vs. Reductionism). This, however, needs to be verified further.

4.2 Chemical kinetics

To asses, if the chemical kinetics as calculated by the implementation of the suggested model correspond to the continuous solutions presented in literature [Cornish-Bowden, 2013, Capellos and Bielski, 1980], different model GAs have been designed. If applicable, the concrete continuous solution has been calculated at every point, where the GA

system also produces a partial solution (after every time step). In the case of second-order kinetics the half-life time was used to compare both methods, since the integral form of second-order reactions have to be considered individually for different rate orders – and therefore are hard to come by. The progression of the MICHAELIS-MENTEN kinetics has been simulated with the software COPASI since exact analytical solutions of the MICHAELIS-MENTEN equation are also difficult to obtain.

4.2.1 First-order kinetics

To evaluate the discrete form of first-order kinetics as they have been introduced in Section 1.2.3 a graph automaton has been constructed. This GA only contains one node and no edges, since diffusion is not relevant in this case. The following reaction for the decomposition of Dinitrogen Pentoxide [Brauer, 2012] will be simulated:

$$2\,N_2O_5 \rightarrow 4\,NO_2 + O_2$$

As species Dinitrogen Pentaoxide, Nitrogen Dioxide and Dioxygen (Oxygen gas) have been added to the automaton. The starting concentration of Dinitrogen Pentaoxide has been set to $0.02\ {}^{mol}/_{l}$ while both products have been set to a concentration of $0\ {}^{mol}/_{l}$. The rate coefficient k for this reaction has been set to $0.07\ {}^{1}/_{s}$. The system has been simulated with a time step of $1\ ms$ and $10\ ms$, respectively, to evaluate whether the dimensional scaling has an impact on the accuracy of the prediction. The progression of the reaction over the course of the simulation can be seen in Figure 4.6.

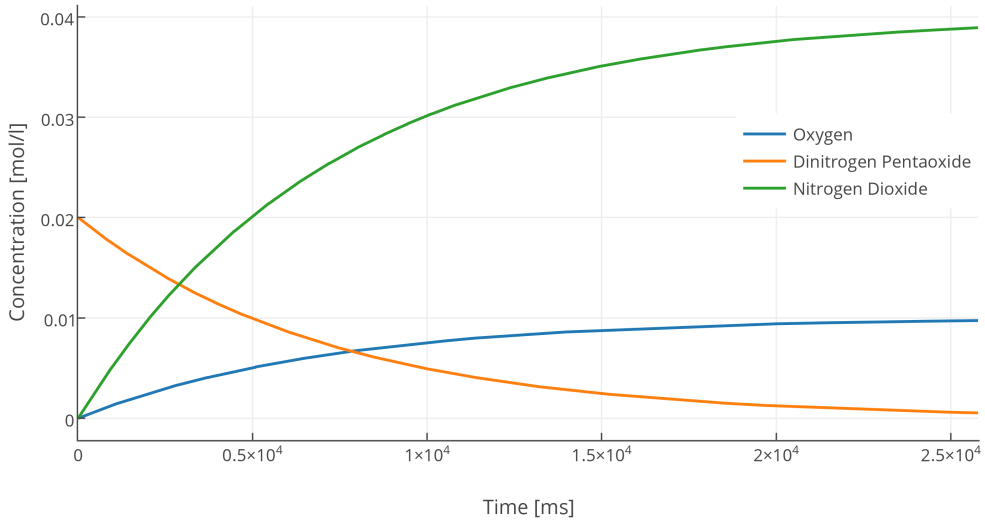


Figure 4.6: Depicted is the progression of the decomposition of dinitrogen pentoxide. The change in concentration of the different reaction species is shown over the first $25\ s$.

The Equation 1.11 can be used to calculate the concentration at any point of the sim-

ulation. To compare both strategies, the continuous solution has been calculated for every point t_i where the simulation has also calculated a result. Both strategies have been found to be congruent for all results. Either of the simulations with time steps of 1 ms and 10 ms, respectively, have a maximal error less than 0.05% (data not shown) per result.

4.2.2 Second-order kinetics

To evaluate the discrete form of second-order kinetics, a graph automaton has been constructed. The reaction for the synthesis of 1,3,5-octatriene (C_8H_{12}) from Buta-1,3-diene (C_4H_6) [Kim, 2004] will be simulated:

$$2\,C_4H_6 \rightarrow C_8H_{12}$$

Both species 1,3,5-octatriene and Buta-1,3-diene have been added to the automaton. The starting concentration of buta-1,3-diene has been set to 0.01 $^{mol}/_{l}$ and the products concentration has been set to 0 $^{mol}/_{l}$. The rate coefficient k for this reaction has been estimated by to 0.0614 $^{l}/_{mol \cdot s}$. The system has been simulated with a time step of 1 s for $10,000$ seconds (about 166 minutes). The progression of the reaction over the course of the simulation can be seen in Figure 4.7.

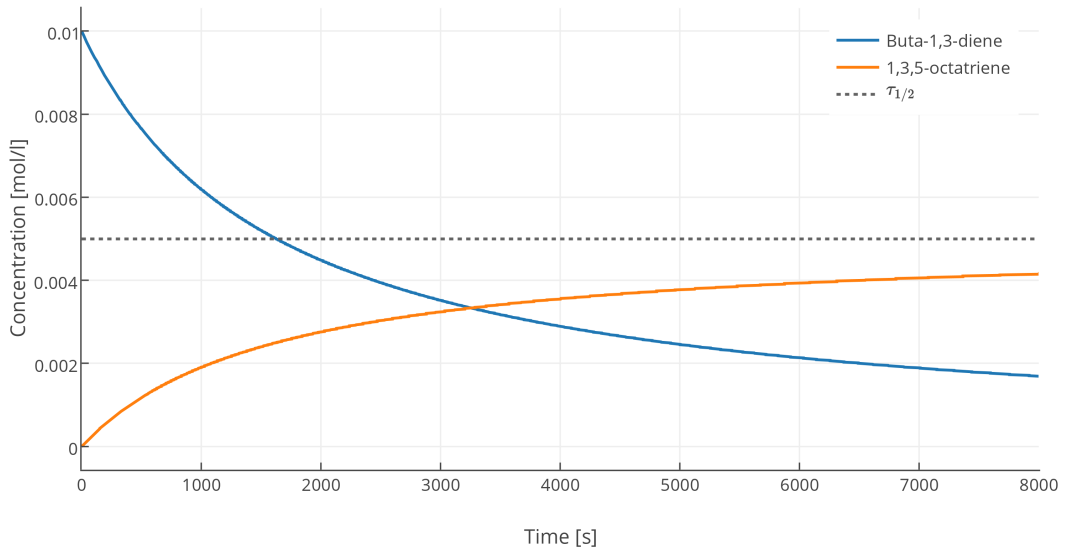


Figure 4.7: Depicted is the progression of the synthesis of 1,3,5-octatriene (C_8H_{12}) from Buta-1,3-diene (C_4H_6). Plotted is the concentration of the different species over the first $8,000$ s. The half-life time $\tau_{1/2}$ is indicated by the dashed horizontal line.

In this evaluation the half-life time of the simulated system and the calculated solution will be compared. The half-life time has been calculated traditionally with the Equation 1.14, which resulted in $\tau_{1/2} = 1628.66$ s. The half-life time in the simulation could be

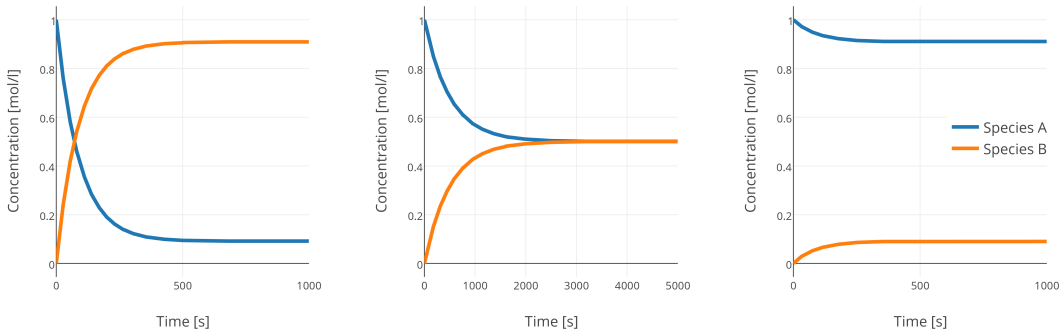
observed after $1627\ s$, with a Buta-1,3-diene concentration of $0.0050014\ {}^{mol}/_{l}$. Again the implementation of the system and the discrete model of chemical kinetics seem to hold very well compared to traditional calculations.

4.2.3 Reversible reactions

A system has been created to evaluate the impact of different equilibrium constants k_{eq}. The system has two generic species A and B that symbolize two reactants that react with each other at an equilibrium:

$$A \rightleftharpoons B$$

Then the system has been simulated three times with different forward and backward reaction rates. The first run used reaction rates of $k_1 = 0.01\ {}^{1}/_{s}$ and $k_{-1} = 0.001\ {}^{1}/_{s}$ resulting in an in an equilibrium constant of $k_{eq} = 10$. In the second run the reaction rates were $k_1 = k_{-1} = 0.01\ {}^{1}/_{s}$, which intuitively resulted in an equilibrium constant of $k_{eq} = 1$. The last run used $k_1 = 0.001\ {}^{1}/_{s}$ and $k_{-1} = 0.01\ {}^{1}/_{s}$ which led to an equilibrium constant of $k_{eq} = 0.1$. All three runs were initialized with a starting concentration of $c(A) = 1.0\ {}^{mol}/_{l}$ and $c(B) = 0.0\ {}^{mol}/_{l}$ and have been simulated for $5,000\ s$ with a timestep of $1\ s$. The resulting progression is depicted in Figure 4.8.



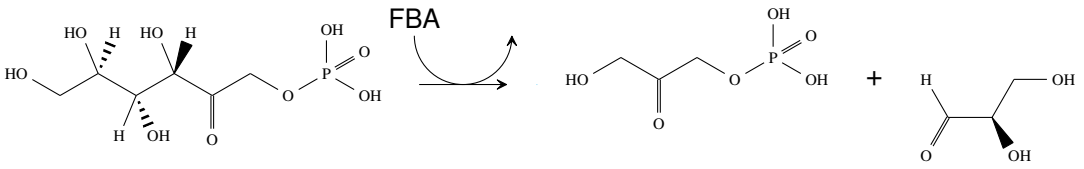
(a) k_{eq} = 10 (b) k_{eq} = 1 (c) k_{eq} = 0.1

Figure 4.8: Examples of simulations with different equilibrium constants. (a) With k_{eq} = 10 the products prevail the substrates tenfold. (b) At k_{eq} = 1 the concentrations of substrates and products are equivalent. (c) The concentrations of substrates outweigh the products tenfold at k_{eq} = 0.1.

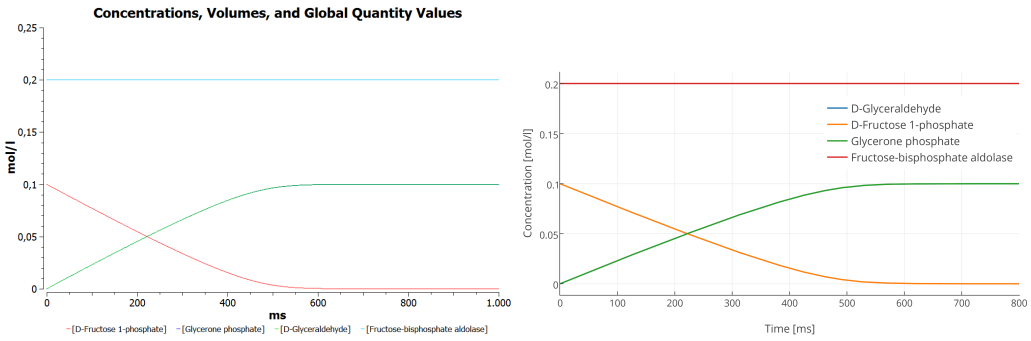
Additionally the systems where calculated with the Equation 1.16 at every time t. After comparing the solutions, they only diverge about 0.5% at the start of the reaction but the error approaches 0 when $t \to \infty$. The behaviour of both species is as predicted with the continuous solution, and the accuracy of the solution is very good.

4.2.4 Michaelis-Menten kinetics

As an exemplary validation of the implemented model of the MICHAELIS-MENTEN kinetics, a sample system with the enzyme Fructose Bisphosphate Aldolase has been simulated once with GASi and once with COPASI [Hoops et al., 2006]. The enzyme Fructose Bisphosphate Aldolase (FBA) ses D-Fructose 1-phosphate to Glycerone Phosphate and D-glyceraldehyde:



Substrate, products, and the enzyme have been added to the automaton. The starting concentration of D-Fructose 1-phosphate have been set to $0.1\ {}^{mol}/_{l}$, the products concentrations has been set to $0\ {}^{mol}/_{l}$ and the enzyme concentration have been set to $0.2\ {}^{mol}/_{l}$. The turnover number k_{cat} for this reaction has been calculated by CALLENS ET AL. to equal $76\ {}^{1}/_{min}$ as well as the value $k_m = 0.009\ {}^{mol}/_{l}$ [Callens et al., 1991]. The system has been simulated with a time step of $1\ ms$ for $1,000\ ms$. The same parameters have been defined for the simulation in COPASI, witch uses the LSODA [Petzold, 1983] method to calculate a numerical approximation of partial differential equations. The resulting simulation progressions can be seen in Figure 4.9. The comparison between both methods shows no significant deviation between the courses of the concentrations. However, the COPASI simulation is approaching its steady state with a slightly steeper course. Still this variance constitutes to a maximal divergence of $4.1 \cdot 10^{-5}\ {}^{mol}/_{l}$ between the both predictions of the substrate concentrations.



(a) Simulation with COPASI (b) Simualtion with GASi

Figure 4.9: The reaction catalysed by FBA has been simulated once with COPASI (a) and once with GASi (b). Since both products increase similarly, both lines are layered on top of each other.

4.3 Advantages and limitations of graph automata

It was shown that RR are able to predict chemical kinetics as good as DE. Diffusion processes show promising results but need to be refined. Considering that only the weight of the species is given to approximate the diffusion coefficient and eventually the process of diffusion, a deviation of 20% in comparison to the experiment is reasonable. The system is easily extensible for different kinds of reactions and for data integration. Different scales from nano-seconds to minutes and nano-metres to centimetres are applicable. Comparably little setup time is needed because many parameters are approximated or imported from different databases. And finally, the state of the system can be observed and altered at any time to provide a better understanding of the processes and their synergy.

On the contrary, there are many strong systems on the market (VCell amongst others) that are already established with a good user base, plentiful resources and very accurate predictions. This makes it hard to build a new software from the ground up, that is able to compete and acquire popularity. Nearly all equations to calculate the kinetic processes and other phenomena are given as differential equations. Although the process is simple, each equation has to be verified and implemented as a reaction scheme in order to incorporate them into the discrete GA system. GA are only able to predict a system, when enough parameters to describe the biological phenomenon are available. These properties need to be calculated, measured or predicted via experimental observations or molecular dynamic simulations. GA are not suitable to predict those parameters in their current form, but are also not designed to do so. Another shortcoming of GA can be that every intermediate state has to be calculated, where a continuous solution can easily be calculated at the desired point in time. This can lead to a computational overhead when only the end result (e.g. the equilibrium state) is required.

5 Further directions

There are generally three sections where improvements and further work can be attached. Those are the mathematical model, the incorporation of other biological aspects, and the user-program-interface.

The mathematical model could be refined by putting more focus on the edges. Currently the length of an edge has no impact on the system. It could be possible to refine a certain area of the graph, such that more nodes are in an defined area. The length of the edges (the node distance) would than be defined dynamically in the system and the scalability would be even greater. Important processes that are very sensitive to spacial scale could be carried by many nodes, where as regions of lower complexity can be approximated with a small amount of nodes. Another aspect of edges could be the definition of a permeability that clarifies if the diffusion was hindered in a certain part of the system (e.g. compartment membranes or aggregates of filaments).

More and potent kinetic equations like the HILL equation are important to model many pathways that are not explainable with simple reversible reactions and MICHAELIS-MENTEN kinetics [Hofmeyr and Cornish-Bowden, 1997]. Those reaction types need to be implemented and evaluated versus experiments. The best solution to improve the diffusion processes would be to conduct experiments with standardised solutions, solutes, and temperatures via a high throughput methods. The results should than be used to provide a reliable correlation between molecular mass and diffusion coefficient by multidimensional curve fitting. An important aspect of self replicating systems such as cells is multiplication of information. The mathematical model should be able to provide the basis to define a mechanism that enables the replication of DNA, its transcription to RNA, and finally the translation to proteins. This feature would be a huge milestone on the way of building a whole cell simulation environment.

A thorough testing of the API and the application is the next step to refine the implementation of the model. More tools for data integration and data export are necessary to share and incorporate resources in the era of big data. The user experience can be improved by the design of wizards that guide through the process of building a working system. Because a continuous development of the software is intended, the latest version of the GASi software will be maintained on `https://bitbucket.org/cleberecht/gasim`.

The graph automata approach provides a scalable framework to add different modules to the overall simulation system. In the future this could be the basis to build a framework

that helps to entangle the complexity of processes that occur inside of a cell.

Appendix A: Appendix

A.1 Tables

Table A.1: Listed are the molar volumes, molar mass, and diffusivity of 86 small molecular species. The molar masses of the substances have been fetched from the Chemical Entities of Biological Interest (ChEBI) database [Degtyarenko et al., 2008]. The molar volumes have been taken from [Hayduk and Laudie, 1974]. Molar mass is given in $^{g}/_{mol}$, molar volume in $^{cm^3}/_{g \cdot mol}$, and diffusivity in $^{cm^2}/_{s}$

Species	Molar mass	Molar volume	Diffusivity
H_2	2.0159	28.5	4.40E-05
He	4.0026	31.9	6.28E-05
NO	30.0061	23.6	2.34E-05
N_2O	46.0055	36.0	2.10E-05
N_2	28.0134	34.7	1.99E-05
NH_3	17.0305	24.5	2.28E-05
O_2	31.9988	27.9	2.29E-05
CO	28.0101	34.5	2.30E-05
CO_2	44.0095	37.3	1.92E-05
SO_2	64.0638	43.8	1.59E-05
H_2S	34.0809	35.2	2.10E-05
Cl_2	70.9060	45.5	1.48E-05
Ar	39.9480	29.2	1.98E-05
Kr	83.7980	34.9	1.92E-05
Ne	20.1797	16.8	3.01E-05
CH_4	16.0425	37.7	1.67E-05
CH_3Cl	50.4875	50.6	1.49E-05
C_2H_3Cl	62.4982	64.5	1.34E-05
C_2H_6	30.0690	53.5	1.38E-05
C_2H_4	28.0532	49.4	1.55E-05
C_3H_8	44.0956	74.5	1.16E-05
C_3H_6	42.0797	69.0	1.44E-05
C_4H_{10}	58.1222	96.6	9.70E-06
Pentane	72.1488	118.0	9.70E-06
Hexane	86.1754	117.0	9.00E-06
2-Methylpentan	86.1754	120.0	9.30E-06
Benzene	78.1118	96.5	1.09E-05

Species	Molar mass	Molar volume	Diffusivity
Toluene	92.1384	118.0	9.50E-06
E-benzene	106.1650	140.0	9.00E-06
Methanol	32.0419	42.5	1.66E-05
Ethanol	46.0684	62.6	1.24E-05
Propanol	60.0950	81.8	1.12E-05
Isopropanol	60.0950	81.8	1.08E-05
Butanol	74.1216	101.0	9.80E-06
Isobutanol	74.1216	101.0	9.30E-06
Benzyl alcohol	108.1378	121.0	9.30E-06
Ethylene glycol	62.0678	63.5	1.16E-05
Triethylen glycol	150.1730	163.0	7.60E-06
Propylene glycol	76.0944	84.5	1.00E-05
Glycerol	92.0938	85.1	9.30E-06
Ethane-1.2-diol	230.3868	197.0	6.40E-06
Acetone	58.0791	77.5	1.28E-05
Ethyl acetate	88.1051	106.0	1.12E-05
Furfural	96.0841	97.5	1.12E-05
Water (self)	18.0152	18.8	2.45E-05
Urea	60.0553	58.0	1.38E-05
Formamide	45.0406	46.6	1.67E-05
Acetamide	59.0672	67.1	1.32E-05
Propionamide	73.0938	87.4	1.20E-05
Butyramide	87.1204	109.0	1.07E-05
Isobutyramide	87.1204	103.0	1.02E-05
Diethylamine	73.1368	109.0	1.11E-05
Aniline	93.1265	106.0	1.05E-05
Formic acid	46.0254	41.1	1.52E-05
Acetic acid	60.0520	64.1	1.19E-05
Propionic acid	74.0785	85.3	1.01E-05
Butyric acid	88.1051	108.0	9.20E-06
Isobutyric acid	88.1051	109.0	9.50E-06
Valeric acid	102.1317	129.0	8.20E-06
Pivalic acid	102.1317	127.0	8.20E-06
Chloracetic acid	94.4970	79.6	1.04E-05
Glycolic acid	76.0514	71.2	9.80E-06
Hexanoic Acid	116.1583	153.0	7.80E-06
Succinic acid	118.0880	120.0	8.60E-06
Glutaric acid	132.1146	142.0	7.90E-06
Adipic acid	146.1412	165.0	7.40E-06
Pimelic acid	160.1678	187.0	7.10E-06
Glycine	75.0666	78.0	1.06E-05
Alanine	89.0932	100.0	9.10E-06

Species	Molar mass	Molar volume	Diffusivity
Serine	105.0926	108.0	8.80E-06
Aminobutyric acid	103.1198	122.0	8.30E-06
Valine	117.1463	145.0	7.70E-06
Leucine	131.1729	167.0	7.30E-06
Proline	115.1305	127.0	8.80E-06
3-Hydroxyproline	131.1299	135.0	8.30E-06
Histidine	155.1546	168.0	7.30E-06
Phenylalanine	165.1891	189.0	7.05E-06
Tryptophan	204.2252	223.0	6.60E-06
Glycylglycine	132.1179	138.0	7.91E-06
Triglycine	189.1692	198.0	6.65E-06
Glycylleucine	188.2242	227.0	6.23E-06
Leucylglycylglycine	245.2756	287.0	5.51E-06
2-Aminobenzoic acid	137.1360	144.0	8.40E-06
Glucose	180.1559	166.0	6.75E-06
Sucrose	342.2965	325.0	5.24E-06
Raffinose	504.4371	480.0	4.34E-06

Table A.2: This table contains the molar mass and diffusivity of 43 proteins. The diffusion coefficients have been taken from [Young et al., 1980]. The molar masses of the proteins have been fetched from the Universal Protein Resource (UniProt) database [UniProt Consortium, 2008] under consideration of the natural subunit structure. Molar mass is given in g/mol and diffusivity in cm^2/s

Protein	**UniProt ID(s)**	**Organism**	**State**	**Molar mass**	**Diffusivity**
Ribonuclease pancreatic	P61823	Bos taurus	monomer	16461	1.02E-07
Alpha-Lactalbumin	P00711	Bos taurus	monomer	16247	1.06E-07
Lysozyme C	P00698	Gallus gallus	monomer	16239	1.12E-07
Myoglobin	P02185	Physeter catodon	monomer	17331	1.13E-07
Myoglobin	P68082	Equus caballus	monomer	17083	1.13E-07
Myoglobin	P02144	Homo sapiens	monomer	17184	1.04E-07
Chymotrypsin A	P00766	Bos taurus	monomer	25666	1.02E-07
Chymotrypsin B	P00767	Bos taurus	monomer	25755	9.90E-08
beta-Casein	P02666	Bos taurus	monomer	25107	6.05E-08
Chymotrypsin C	Q7M3E1	Bos taurus	monomer	29255	9.30E-08
Riboflavin-binding protein	P02752	Gallus gallus	monomer	27211	7.40E-08
Pepsin A	P00791	Sus scrofa	monomer	41262	9.00E-08
Beta-lactoglobulin	P02754	Bos taurus	homodimer	39766	7.80E-08
Flagellin	P04949	Escherichia coli	monomer	51295	5.40E-08
Ovalbumin	P01012	Gallus gallus	monomer	42881	7.76E-08
Phosphoglycerate kinase	P00560	Saccharomyces cerevisiae	monomer	44738	6.38E-08
Hemoglobin	P69905 P68871	Homo sapiens	heterotetramer	62512	6.30E-08
Hemoglobin	P01958 P02062	Equus caballus	heterotetramer	62506	6.30E-08
Serum albumin	P02769	Bos taurus	monomer	69293	5.81E-08

Protein	UniProt ID(s)	Organism	State	Molar mass	Diffusivity
Hexokinase	P04806 P04807	Saccharomyces cerevisiae	heterodimer	107680	5.90E-08
Myosin light chain	Q28824	Bos taurus	monomer	128825	2.25E-08
Lysine-tRNA ligase	P15180	Saccharomyces cerevisiae	homodimer	135918	4.30E-08
Lactate dehydrogenase M4	P00341	Squalus acanthias	homotetramer	146780	5.10E-08
Glyceraldehyde dehydrogenase	P00360	Saccharomyces cerevisiae	homotetramer	143000	4.90E-08
Phosphofructokinase	P0A769	Escherichia coli	homotetramer	139368	5.30E-08
Phosphofructokinase	P00511	Oryctolagus cuniculus	homodimer	170406	4.20E-08
Beta-lactoglobulin	P02754	Bos taurus	homooctamer	159064	4.20E-08
Immunoglobulin G	Q8N6C5	Homo sapiens	monomer	148936	4.00E-08
Methionyl-tRNA ligase	P00959	Escherichia coli	homodimer	152510	3.50E-08
Glycogen phosphorylase	P00489	Oryctolagus cuniculus	homodimer	194578	4.12E-08
Malate synthese	P21826	Saccharomyces cerevisiae	homotrimer	188382	4.50E-08
Pyruvate kinase	P00549	Saccharomyces cerevisiae	homotetramer	218180	4.20E-08
Catalase	P00432	Bos taurus	homotetramer	239660	4.10E-08
Aminolevulinic acid dehydratase	P10518	Mus musculus	homooctamer	288192	4.20E-08
Glutamate dehydrogenase	P00366	Bos taurus	homohexamer	375072	3.50E-08
Phosphofructokinase	P08237	Homo sapiens	homotetramer	340732	3.60E-08
Collagen	P02458	Homo sapiens	homotrimer	425355	6.80E-09
Fibrinogen	P02671 P02675 P02679	Homo sapiens	heterohexamer	404826	2.00E-08
Phosphofructokinase	P00511	Oryctolagus cuniculus	homotetramer	340812	3.22E-08
Adenovirus Hexon Protein	P03277	Human adenovirus C serotype 2	homotrimer	327459	3.56E-08

Protein	UniProt ID(s)	Organism	State	Molar mass	Diffusivity
Glycogen phosphorylase	P00489	Oryctolagus cuniculus	homotetramer	389156	3.50E-08
Apofemtin	P02791	Equus caballus	oligomer (24)	479472	3.61E-08
Thyroglobulin	P01267	Bos taurus	homodimer	606444	2.65E-08

A.2 Poster

A poster (see Figure A.1) has been created and presented at the Centre for Regenerative Therapies Dresden (CRTD) during the 9th CRTD Summer Conference on Regenerative Medicine on June 5, 2015.

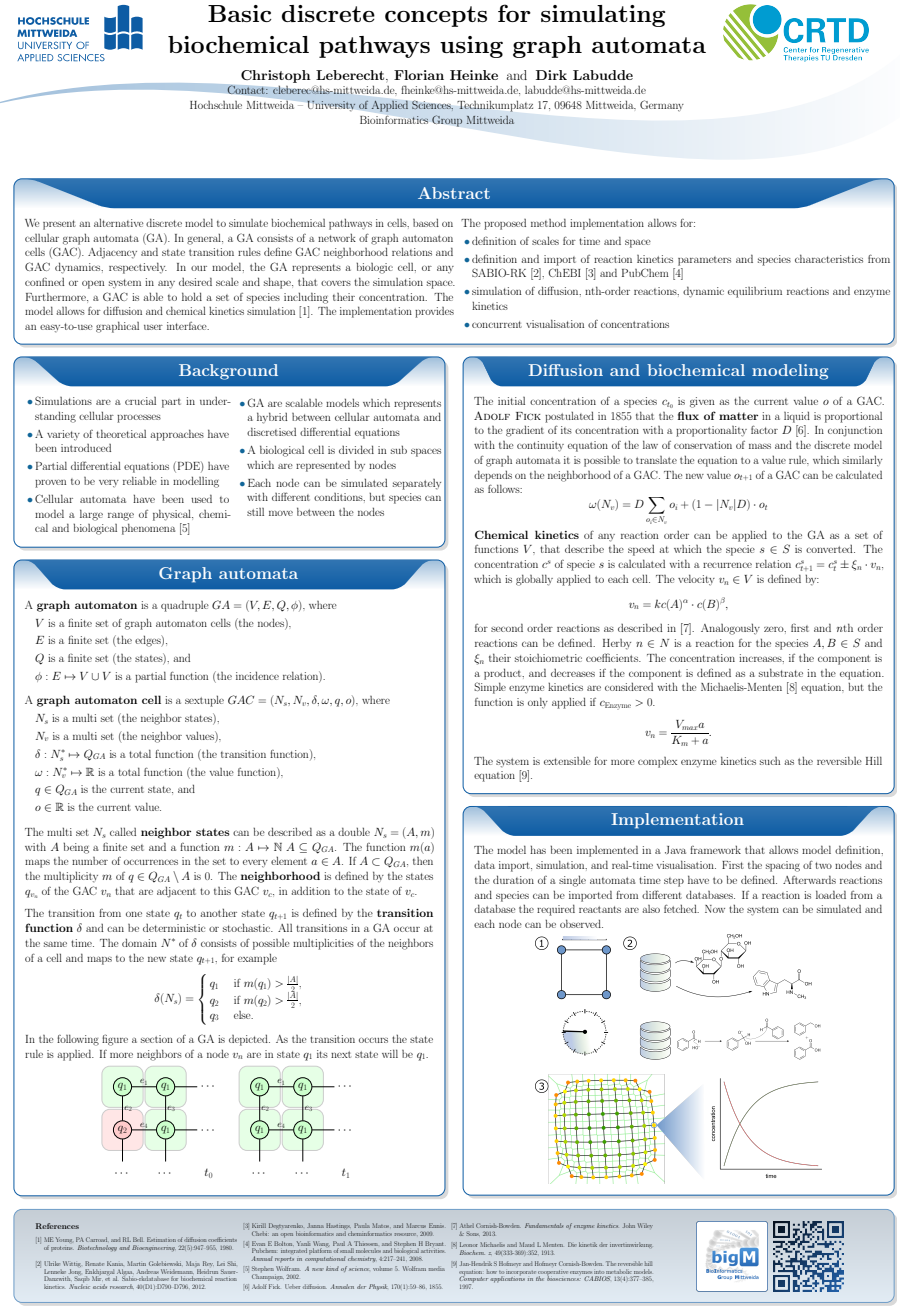


Figure A.1: CRTD Poster

A.3 CD content

The enclosed CD contains the following material:

- this thesis in pdf format
- all figures in high resolution
- the poster A.1 in A0 format
- the developed software as a Java project

Bibliography

[Adami, 1998] Adami, C. (1998). *Introduction to artificial life*, volume 1. Springer Science & Business Media.

[Anderson, 1973] Anderson, P. (1973). More is different - broken symmetry and the nature of the hierarchical structure of science. *Science*, 177:393–396.

[Atkins and De Paula, 1998] Atkins, P. W. and De Paula, J. (1998). *Physical chemistry*. Oxford University Press, 4 edition.

[Auyang, 1999] Auyang, S. Y. (1999). *Foundations of complex-system theories: In economics, evolutionary biology, and statistical physics*. Cambridge University Press.

[Bolton et al., 2008] Bolton, E. E., Wang, Y., Thiessen, P. A., and Bryant, S. H. (2008). Pubchem: Integrated platform of small molecules and biological activities. *Annual Reports in Computational Chemistry*, 4:217–241.

[Brandes et al., 2002] Brandes, U., Eiglsperger, M., Herman, I., Himsolt, M., and Marshall, M. S. (2002). Graphml progress report structural layer proposal. In *Graph Drawing*. Springer.

[Brauer, 2012] Brauer, G. (2012). *Handbook of preparative inorganic chemistry*, volume 2. Elsevier. 489–491.

[Briggs and Haldane, 1925] Briggs, G. E. and Haldane, J. B. S. (1925). A note on the kinetics of enzyme action. *Biochemical Journal*, 19(2):338.

[Brin and Stuck, 2002] Brin, M. and Stuck, G. (2002). *Introduction to dynamical systems*. Cambridge University Press.

[Brown, 1892] Brown, A. J. (1892). Influence of oxygen and concentration on alcoholic fermentation. *Journal of the Chemical Society, Transactions*, 61:369–385.

[Buchner, 1897] Buchner, E. (1897). Alkoholische gärung ohne hefezellen. *Berichte der deutschen chemischen Gesellschaft*, 30(1):1110–1113.

[Callens et al., 1991] Callens, M., Kuntz, D. A., and Opperdoes, F. R. (1991). Kinetic properties of fructose bisphosphate aldolase from trypanosoma brucei compared to

aldolase from rabbit muscle and staphylococcus aureus. *Molecular and Biochemical Parasitology*, 47(1):1–9.

[Capellos and Bielski, 1980] Capellos, C. and Bielski, B. H. J. (1980). *Kinetic systems: Mathematical description of chemical kinetics in solution*. Krieger Publishing Company.

[Charlwood, 1957] Charlwood, P. (1957). Partial specific volumes of proteins in relation to composition and environment. *Journal of the American Chemical Society*, 79(4):776–781.

[Collett et al., 2015] Collett, C. J., Massey, R. S., Taylor, J. E., Maguire, O. R., O'Donoghue, A. C., and Smith, A. D. (2015). Rate and equilibrium constants for the addition of n-heterocyclic carbenes into benzaldehydes: A remarkable 2-substituent effect. *Angewandte Chemie*, 127(23):6991–6996.

[Cornish-Bowden, 2013] Cornish-Bowden, A. (2013). *Fundamentals of enzyme kinetics*. John Wiley & Sons.

[de Lavoisier, 1801] de Lavoisier, A.-L. (1801). *Traité élémentaire de chimie*. Deterville.

[Degtyarenko et al., 2008] Degtyarenko, K., De Matos, P., Ennis, M., Hastings, J., Zbinden, M., McNaught, A., Alcántara, R., Darsow, M., Guedj, M., and Ashburner, M. (2008). Chebi: A database and ontology for chemical entities of biological interest. *Nucleic Acids Research*, 36(suppl 1):D344–D350.

[Dohmen, 2013] Dohmen, K. (2013). *Automatentheorie*. Lecture series, HS Mittweida.

[Duffett-Smith, 1992] Duffett-Smith, P. (1992). *Time. Practical astronomy with Your Calculator*. Cambridge University Press, 3rd edition.

[Eggleston and Hems, 1952] Eggleston, L. and Hems, R. (1952). Separation of adenosine phosphates by paper chromatography and the equilibrium constant of the myokinase system. *Biochemical Journal*, 52(1):156.

[Einstein, 1905] Einstein, A. (1905). Ueber die von der molekularkinetischen theorie der waerme geforderte bewegung von in ruhenden fluessigkeiten suspendierten teilchen. *Annalen der Physik*, 4:549–560.

[Endy and Brent, 2001] Endy, D. and Brent, R. (2001). Modelling cellular behaviour. *Nature*, 409(6818):391–395.

[Érdi and Tóth, 1989] Érdi, P. and Tóth, J. (1989). *Mathematical models of chemical re-*

actions: theory and applications of deterministic and stochastic models. Manchester University Press.

[Farina and Dennunzio, 2008] Farina, F. and Dennunzio, A. (2008). A predator-prey cellular automaton with parasitic interactions and environmental effects. *Fundamenta Informaticae*, 83(4):337–353.

[Fick, 1855] Fick, A. (1855). Ueber diffusion. *Annalen der Physik*, 170(1):59–86.

[Fortune, 1987] Fortune, S. (1987). A sweepline algorithm for voronoi diagrams. *Algorithmica*, 2(1-4):153–174.

[Fruchterman and Reingold, 1991] Fruchterman, T. M. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164.

[Fulton, 1982] Fulton, A. B. (1982). How crowded is the cytoplasm? *Cell*, 30(2):345–347.

[Gaylord and Nishidate, 2013] Gaylord, R. J. and Nishidate, K. (2013). *Modeling nature: Cellular automata simulations with Mathematica*. Springer.

[Guldberg and Waage, 1864] Guldberg, C. M. and Waage, P. (1864). Studies concerning affinity. *CM Forhandlinger: Videnskabs-Selskabet i Christiana*, 35(1864):1864.

[Hayduk and Laudie, 1974] Hayduk, W. and Laudie, H. (1974). Prediction of diffusion coefficients for nonelectrolytes in dilute aqueous solutions. *AIChE Journal*, 20(3):611–615.

[Henri, 1903] Henri, V. (1903). *Lois générales de l'action des diastases*. Librairie Scientifique A. Hermann.

[Hertz, 1863] Hertz, H. (1863). *Die Prinzipien der Mechanik*. Wissenschaftliche Buchgesellschaft, Darmstadt.

[Hofmeyr and Cornish-Bowden, 1997] Hofmeyr, J.-H. S. and Cornish-Bowden, H. (1997). The reversible hill equation: How to incorporate cooperative enzymes into metabolic models. *Computer applications in the biosciences: CABIOS*, 13(4):377–385.

[Hoops et al., 2006] Hoops, S., Sahle, S., Gauges, R., Lee, C., Pahle, J., Simus, N., Singhal, M., Xu, L., Mendes, P., and Kummer, U. (2006). Copasi - a complex pathway simulator. *Bioinformatics*, 22(24):3067–3074.

[Hopcroft et al., 2002] Hopcroft, J. E., Ullman, J. D., and Motwani, R. (2002). *Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie*, volume 2. Pearson Studium Deutschland, München.

[Hütt, 2001] Hütt, M.-T. (2001). *Datenanalyse in der Biologie: Eine Einführung in Methoden der nichtlinearen Dynamik, fraktalen Geometrie und Informationstheorie*. Springer-Verlag.

[Ishii et al., 2004] Ishii, N., Robert, M., Nakayama, Y., Kanai, A., and Tomita, M. (2004). Toward large-scale modeling of the microbial cell for computer simulation. *Journal of Biotechnology*, 113(1):281–294.

[Jamtveit and Meakin, 1999] Jamtveit, B. and Meakin, P. (1999). *Growth, dissolution and pattern formation in geosystems*. Springer.

[Kang et al., 2012] Kang, M., Day, C. A., Kenworthy, A. K., and DiBenedetto, E. (2012). Simplified equation to extract diffusion coefficients from confocal frap data. *Traffic*, 13(12):1589–1600.

[Kao et al., 1993] Kao, H. P., Abney, J. R., and Verkman, A. (1993). Determinants of the translational mobility of a small solute in cell cytoplasm. *The Journal of Cell Biology*, 120(1):175–184.

[Kim, 2004] Kim, C. D. (2004). *Chemical Kinetics (Chapter 12)*. Lecture series, San Joaquin Delta College.

[Kitano, 2002] Kitano, H. (2002). Systems biology: A brief overview. *Science*, 295(5560):1662–1664.

[Koenig, 2012] Koenig, R. (2012). *Simulation und Visualisierung der Dynamik räumlicher Prozesse*. VS Verlag für Sozialwissenschaften.

[Krause and Nesemann, 2012] Krause, U. and Nesemann, T. (2012). *Differenzengleichungen und diskrete dynamische Systeme: Eine Einführung in Theorie und Anwendungen*. Walter de Gruyter.

[Le Chatelier and Boudouard, 1898] Le Chatelier, H. and Boudouard, O. (1898). Limits of flammability of gaseous mixtures. *Bulletin de la Société Chimique de France*, 19:483–488.

[Lévi, 1927] Lévi, R. (1927). Théorie de l'action universelle et discontinue. *Journal de Physique et Le Radium*, 8(4):182–198.

[Lodish et al., 2000] Lodish, H., Berk, A., Lawrence, Z., Matsudaira, P., Baltimore, D., and Darnell, J. (2000). *Molecular cell biology*. Freeman, W. H., 4th edition.

[Luby-Phelps, 2000] Luby-Phelps, K. (2000). Cytoarchitecture and physical properties of cytoplasm: Volume, viscosity, diffusion, intracellular surface area. *International Review of Cytology*, 192:189–221.

[Luby-Phelps et al., 1993] Luby-Phelps, K., Mujumdar, S., Mujumdar, R., Ernst, L., Galbraith, W., and Waggoner, A. (1993). A novel fluorescence ratiometric method confirms the low solvent viscosity of the cytoplasm. *Biophysical Journal*, 65(1):236.

[Madore and Freedman, 1983] Madore, B. F. and Freedman, W. L. (1983). Computer simulations of the Belousov-Zhabotinsky reaction. *Science*, 222(4624):615–616.

[Margenau, 1950] Margenau, H. (1950). *The nature of physical reality: A philosophy of modern physics*. Ox Bow Press.

[Mattheij et al., 2005] Mattheij, R. M., Rienstra, S. W., and ten Thije Boonkkamp, J. H. (2005). *Partial differential equations: Modeling, analysis, computation*. Siam.

[McAdams and Shapiro, 2009] McAdams, H. H. and Shapiro, L. (2009). System-level design of bacterial cell cycle control. *FEBS letters*, 583(24):3984–3991.

[McNaught and Wilkinson, 1997] McNaught, A. D. and Wilkinson, A. (1997). *IUPAC. Compendium of chemical terminology*. Blackwell Scientific Publications, Oxford.

[Mehrer and Stolwijk, 2009] Mehrer, H. and Stolwijk, N. A. (2009). Heroes and highlights in the history of diffusion. *Diffusion Fundamentals*, 11(1):1–32.

[Metzler and Klafter, 2004] Metzler, R. and Klafter, J. (2004). The restaurant at the end of the random walk: Recent developments in the description of anomalous transport by fractional dynamics. *Journal of Physics A: Mathematical and General*, 37(31):R161.

[Michaelis and Menten, 1913] Michaelis, L. and Menten, M. L. (1913). Die kinetik der invertinwirkung. *Biochemische Zeitschrift*, 49(352):333–369.

[Omenn, 2006] Omenn, G. S. (2006). Grand challenges and great opportunities in science, technology, and public policy. *Science*, 314(5806):1696–1704.

[O'Sullivan and Tompson, 1890] O'Sullivan, C. and Tompson, F. W. (1890). Invertase: A contribution to the history of an enzyme or unorganised ferment. *Journal of the Chemical Society, Transactions*, 57:834–931.

[Pagliaro and Taylor, 1992] Pagliaro, L. and Taylor, D. L. (1992). 2-deoxyglucose and cytochalasin d modulate aldolase mobility in living 3t3 cells. *The Journal of cell biology*, 118(4):859–863.

[Petzold, 1983] Petzold, L. (1983). Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. *SIAM Journal on Scientific and Statistical Computing*, 4(1):136–148.

[Planck, 1943] Planck, M. (1943). Zur geschichte der auffindung des physikalischen wirkungsquantums. *Naturwissenschaften*, 31(14):153–159.

[R Development Core Team, 2008] R Development Core Team (2008). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria.

[Reits and Neefjes, 2001] Reits, E. A. and Neefjes, J. J. (2001). From fixed to frap: measuring protein mobility and activity in living cells. *Nature Cell Biology*, 3(6):E145–E147.

[Sitaraman et al., 1963] Sitaraman, R., Ibrahim, S., and Kuloor, N. (1963). A generalized equation for diffusion in liquids. *Journal of Chemical and Engineering Data*, 8(2):198–201.

[Slepchenko et al., 2003] Slepchenko, B. M., Schaff, J. C., Macara, I., and Loew, L. M. (2003). Quantitative cell biology with the virtual cell. *Trends in Cell Biologyl*, 13(11):570–576.

[Swaminathan et al., 1997] Swaminathan, R., Hoang, C. P., and Verkman, A. (1997). Photobleaching recovery and anisotropy decay of green fluorescent protein gfp-s65t in solution and cells: cytoplasmic viscosity probed by green fluorescent protein translational and rotational diffusion. *Biophysical Journal*, 72(4):1900.

[Thomas, 2002] Thomas, W. (2002). *Automatentheorie und formale Sprachen*. Lecture series, RWTH Aachen.

[UniProt Consortium, 2008] UniProt Consortium (2008). The universal protein resource (uniprot). *Nucleic Acids Research*, 36(suppl 1):D190–D195.

[Verkman, 2002] Verkman, A. S. (2002). Solute and macromolecule diffusion in cellular aqueous compartments. *Trends in Biochemical Sciences*, 27(1):27–33.

[Von Smoluchowski, 1906] Von Smoluchowski, M. (1906). Zur kinetischen theorie der brownschen molekularbewegung und der suspensionen. *Annalen der Physik*, 326(14):756–780.

[Wanjek, 2011] Wanjek, C. (2011). Systems biology as defined by NIH. *NIH Catalyst*, 19(6):10–12.

[West, 2001] West, D. B. (2001). *Introduction to graph theory*, volume 2. Prentice Hall University of Michigan.

[Wilhelmy, 1850] Wilhelmy, L. (1850). Ueber das gesetz, nach welchem die einwirkung der säuren auf den rohrzucker stattfindet. *Annalen der Physik*, 157(12):499–526.

[Wilke and Chang, 1955] Wilke, C. and Chang, P. (1955). Correlation of diffusion coefficients in dilute solutions. *AIChE Journal*, 1(2):264–270.

[Wishart et al., 2005] Wishart, D. S., Yang, R., Arndt, D., Tang, P., and Cruz, J. (2005). Dynamic cellular automata: An alternative approach to cellular simulation. *In Silico Biol*, 5(2):139–161.

[Wittig et al., 2012] Wittig, U., Kania, R., Golebiewski, M., Rey, M., Shi, L., Jong, L., Algaa, E., Weidemann, A., Sauer-Danzwith, H., Mir, S., et al. (2012). Sabio-rk - database for biochemical reaction kinetics. *Nucleic Acids Research*, 40(D1):D790–D796.

[Wolfram, 2002] Wolfram, S. (2002). *A new kind of science*, volume 5. Wolfram Media Champaign.

[Young et al., 1980] Young, M., Carroad, P., and Bell, R. (1980). Estimation of diffusion coefficients of proteins. *Biotechnology and Bioengineering*, 22(5):947–955.

Erklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Mittweida, 20.08.2015